JFW

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICANT:	Hyun-Ju PARK	)
		)
SERIAL NO.	10/589,595	)
		)
DATE FILED:	August 15, 2006	)
		)
FOR:	CHIP DESIGN VERIFICATION APPARATUS AND METHOD	)

SUBMISSION OF CERTIFIED COPY OF FOREIGN APPLICATION UNDER 37 CFR 1.55

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Commissioner:

Enclosed herewith is a certified copy of Korean Patent Application No. 10-2004-0010490 filed on February 17, 2004 in Korea. The enclosed Application is directed to the invention disclosed and claimed in the above-identified application.

Applicants hereby reaffirm the claim of priority previously made on August 15, 2006 claiming the benefit of the filing date of February 17, 2004, of the Korean Patent Application No. 10-2004-0010490, under provisions of 35 U.S.C. 119 and the International Convention for the protection of Industrial. Property.

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450 [37 CFR 1.8(a)] on:

July 16, 2007
(date of deposit)

Paula Currie
(Typed or Printed Name of Person Mailing Correspondence)

(Signature of Person Mailing Correspondence)

Respectfully submitted,

CANTOR COLBURN LLP

By: Amy Bizon-Copp
Amy Bizon-Copp
Registration No. 53,993
Cantor Colburn LLP
55 Griffin Road South
Bloomfield, CT 06002
Telephone: (860) 286-2929
Facsimile: (860) 286-0115
Customer No. 23413

Issue Number : 5-5-2007-034175560

대한민국 특허청
KOREAN INTELLECTUAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Intellectual Property Office.

출 원 번 호 : Application Number		10-2004-0010490
출 원 년 월 일 : Filing Date		2004년 02월 17일 FEB 17, 2004
출 원 인 : Applicant(s)		박현주 PARK HYUN JU

2007년 06월 20일

특 허 청
COMMISSIONER

◆ This certificate was issued by Korean Intellectual Property Office. Please confirm any forgery or alteration of the contents by an issue number or a barcode of the document below through the KIPOnet- Online Issue of the Certificates' menu of Korean Intellectual Property Office homepage (www.kipo.go.kr). But please notice that the confirmation by the issue number is available only for 90 days.

Issue Date : 2007.06.20

【서지사항】

【서류명】 명세서 등 보정서

【수신처】 특허청장

【제출일자】 2004.03.05

【제출인】

【성명】 박현주

【출원인코드】 4-2000-026787-4

【사건과의 관계】 출원인

【대리인】

【성명】 박상수

【대리인코드】 9-1998-000642-5

【포괄위임등록번호】 2000-053697-9

【사건의 표시】

【출원번호】 10-2004-0010490

【출원일자】 2004.02.17

【심사청구일자】 2004.02.17

【발명의 명칭】 칩 설계 검증 장치 및 방법

【제출원인】

【접수번호】 1-1-2004-0065616-02

【접수일자】 2004.02.17

【보정할 서류】 명세서등

【보정할 사항】

【보정대상항목】 별지와 같음

【보정방법】 별지와 같음

【보정내용】 별지와 같음

【취지】 특허법시행규칙 제13조 · 실용신안법시행규칙 제8조의 규정에의하여 위와 같 이 제출합니다.

대리인 박상수 (인)

【수수료】

【보정료】	0 원
【추가심사청구료】	0 원
【기타 수수료】	0 원
【합계】	0 원

【보정서】

【보정대상항목】 요약

【보정방법】 정정

【보정내용】

【요약】

본 발명은 칩 설계 검증 장치 및 방법을 공개한다. 그 칩 설계 검증 방법은 상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이 수단으로 전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 하드웨어 블록의 유효한 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계와, 상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 소프트웨어 블록의 유효한 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계를 구비하는 것을 특징으로 한다. 따라서 칩 설계시에 칩 설계상의 오류를 검출할 수 있도록 하고, 보다 빠른 칩 설계 검증 속도를 제공하여 준다.

【보정대상항목】 대표도

【보정방법】 정정

【보정내용】

【대표도】

도 3

【보정대상항목】 식별번호 2

【보정방법】 정정

【보정내용】

<2> 도 2는 일반적인 소프트웨어 IP 및 타겟을 구비하는 칩의 개념도이다.

【보정대상항목】 식별번호 3

【보정방법】 정정

【보정내용】

<3> 도 3은 본 발명의 칩 설계 검증 장치의 블록도이다.

【보정대상항목】 식별번호 11

【보정방법】 정정

【보정내용】

<11> 오늘날 반도체 설계 자동화와 관련된 각종 EDA(Electronic Design Automation) 도구(tool)의 보급이 확대되고 HDL(Hardware Description Language)을 이용한 설계 방법이 보편화되면서 ASIC(Application Specific Integrated Circuit)의 설계 환경은 크게 개선되었다.

【보정대상항목】 식별번호 12

【보정방법】 정정

【보정내용】

<12> 따라서 설계되는 회로의 크기도 기존의 수만~수십만 게이트 수준에서 근래에는 수백만~수천만 게이트에 이르는 정도의 용량을 설계하기에 이르러 ASIC은 하나의 칩에 시스템을 구현하는 것이 가능한 정도에 이르렀다. 이와 같이 구성된 시스템을 SoC(System on a Chip)라고 한다.

【보정대상항목】 식별번호 13

【보정방법】 정정

【보정내용】

<13> 또한 ASIC 또는 SoC를 설계할 경우, 모든 것을 제로베이스에서 개발하기 보다는 기존에 이미 존재하는 설계 회로의 일부 기능 블록을 변경하거나 추가하는 형태로 진행이 되는 경우가 보다 일반적인 경우이며, 변경되거나 추가되는 기능 블록 역시 기존에 단위 기능 블록 형태로 존재하는 IP(Intellectual Property) 블록을 활용하는 것이 근래의 ASIC 또는 SoC 설계의 추세라고 할 수 있다.

【보정대상항목】 식별번호 17

【보정방법】 정정

【보정내용】

<17> 첫째, 샘플 형태로 전달될 하드웨어 IP가 볼그레이드어레이(BGA) 또는 다수 핀의 ASIC 또는 FPGA 형태인 경우, 사용을 위해서는 하드웨어 IP가 탑재될 새로운 보드를 제작하여야 하고 보드 제작 이후에는 조립 상태의 여하에 따라서는 IP 동작이 불량 현상으로 나타나는 경우가 빈발하다.

【보정대상항목】 식별번호 21

【보정방법】 정정

【보정내용】

<21> 특히, 본 발명에서 관심을 가지는 분야는 이상과 같은 하드웨어 IP가 갖는 단점을 최소화하면서도 소프트웨어 IP의 본연의 장점을 제공하는 소프트웨어 IP를 활용하는 칩의 설계 검증 장치 및 방법에 관한 것과, 수백만~수천만 게이트에 이르는 SoC 급의 ASIC 설계용량에 대응한 고속의 시뮬레이션 가속을 달성하기 위한 칩의 설계 검증 장치 및 방법에 관한 것이다.

【보정대상항목】 식별번호 23

【보정방법】 정정

【보정내용】

<23> 따라서 상기의 칩 설계 검증의 목적을 구현하기 위해서는 칩의 하드웨어 설계 블록 즉 타겟과 소프트웨어 IP 및/또는 테스트벤치로 구성되는 소프트웨어 블럭의 공동 시뮬레이션을 수행하여야 한다.

【보정대상항목】 식별번호 24

【보정방법】 정정

【보정내용】

<24> 종래의 소프트웨어 IP를 이용하는 칩의 설계 검증 장치 또는 시뮬레이션 가속을 위한 칩 설계 검증 장치로는 미국 특허 번호 제6,356,862에 공개된 지연 동기를 이용한 하드웨어와 소프트웨어 상호 검증기가 있을 수 있겠다. 미국 특허 번호 제6,356,862에 공개된 장치는 하드웨어와 소프트웨어 상호 조정 수단을 구비하여 하드웨어 검증기와 소프트웨어 검증기가 상호 동기되어 검증 동작을 수행할 수 있도록 하였다.

【보정대상항목】 식별번호 27

【보정방법】 정정

【보정내용】

<27> 또한 이때의 소프트웨어 IP와 타겟의 칩 설계 검증에 필요한 테스트 패턴들과 멀티 클럭의 클럭 패턴을 제공하는 테스트 벤치는 컴퓨터 시스템 내에 존재하고, 타겟은 컴퓨터 시스템의 외부에 존재하여, 소프트웨어 IP와 타겟은 별도의 인터페이스 수단을 통해 데이터 통신을 수행하여야 했었다.

【보정대상항목】 식별번호 29

【보정방법】 정정

【보정내용】

<29> 그러나 종래의 칩의 설계 검증 장치의 경우에는 테스트벤치로부터 출력되는 각 클럭 신호의 일정 주기를 기반으로 하여 소프트웨어 IP 또는 타겟에서 발생된 출력 데이터를 상대측에 제공하였었다. 이에 데이터 통신 용량이 불필요하게 증가하게 되는 문제점이 발생하였었다.

【보정대상항목】 식별번호 30

【보정방법】 정정

【보정내용】

<30> 또한 종래의 칩의 설계 검증 장치에 있어서, 타겟은 동작에 필요한 클럭(일반적으로 멀티 클럭)을 컴퓨터 시스템 내에 존재하는 테스트 벤치로부터 제공받아야 했다.

【보정대상항목】 식별번호 31

【보정방법】 정정

【보정내용】

<31> 이에 테스트벤치로부터 출력되는 각 클럭의 변화를 인터페이스 수단으로 전송하는 데 있어서 데이터의 전송 지연이 필연적으로 발생하게 되어, 칩 설계 검증을 위한 타겟의 동작 속도가 불필요하게 느려지는 문제점이 있었다.

【보정대상항목】 식별번호 33

【보정방법】 정정

【보정내용】

<33> 본 발명의 다른 목적은 이벤트 기반으로 소프트웨어 IP 및/또는 테스트벤치로 구성되는 소프트웨어 블록과 타겟간의 데이터 송수신이 수행되도록 하여 데이터 전송 속도 및 효율을 높이는 IP 검증 및/또는 시뮬레이션 가속을 목적으로 한 칩의 설계 검증 장치 및 방법을 제공하는데 있다.

【보정대상항목】 식별번호 35

【보정방법】 정정

【보정내용】

<35> 상기 목적들을 달성하기 위한 본 발명의 칩 설계 검증 장치는 적어도 하나 이상의 하드웨어 블록과, 상기 하드웨어 블록과 데이터 통신을 수행하는 적어도 하나 이상의 소프트웨어 블록을 포함하고, 상기 하드웨어 블록과 상기 소프트웨어 블록간의 동작을 검증하는 컴퓨터를 구비하고, 상기 컴퓨터는 상기 하드웨어 블록의 출력데이터를 전송하고, 상기 소프트웨어 블록의 출력 데이터의 유효성 여부를 판단하고, 소프트웨어 블록의 유효한 출력데이터만을 상기 하드웨어 블록으로 인가하는 인터페이스 수단과, 상기 소프트웨어 블록, 상기 하드웨어 블록과 상기 소프트웨어 블록간의 동작을 검증하는 칩 설계 검증 프로그램을 저장하는 저장수단과, 상기 칩 설계 검증 프로그램을 수행하여 상기 소프트웨어 블록의 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 인터페이스 수단을 통해 입력되는 상기 하드웨어 블록의 출력 데이터의 유효성 여부를 판단하고, 하드웨어 블록의 유효한 출력데이터만을 상기 소프트웨어 블록에 인가하도록 하는 제어부를 구비하는 것을 특징

으로 한다.

【보정대상항목】 식별번호 36

【보정방법】 정정

【보정내용】

<36> 상기 목적들을 달성하기 위한 본 발명의 제 1 형태의 칩 설계 검증 방법은 상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 하드웨어 블록의 유효한 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계와, 상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 소프트웨어 블록의 유효한 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계를 구비하는 것을 특징으로 한다.

【보정대상항목】 식별번호 37

【보정방법】 정정

【보정내용】

<37> 상기 목적들을 달성하기 위한 본 발명의 제 2 형태의 칩 설계 검증 방법은 상기 칩 설계 검증 프로그램은 멀티 클럭 설정값을 획득하고, 획득한 멀티 클럭 설

정값을 상기 인터페이스 수단에 제공하고, 상기 멀티 클럭 설정값과 상기 칩 설계 검증 프로그램의 시스템 클럭에 따라 멀티 클럭을 발생하여 상기 소프트웨어 블록에 인가하고, 상기 인터페이스 수단은 상기 멀티 클럭 설정값과 상기 인터페이스 수단의 시스템 클럭에 따라 멀티 클럭을 발생하여 상기 하드웨어 블록에 인가하는 클럭 생성 단계와, 상기 칩 설계 검증 프로그램의 멀티 클럭에 응답하여 동작하는 상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 하드웨어 블록의 유효한 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계와, 상기 인터페이스 수단의 멀티 클럭에 응답하여 동작하는 상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 소프트웨어 블록의 유효한 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계를 구비하는 것을 특징으로 한다.

【보정대상항목】 식별번호 48

【보정방법】 정정

【보정내용】

<48> 확장 슬롯들(170)은 PCI(Peripheral Component Interconnect) 슬롯으로, LAN

카드, 팩스 모뎀 카드 등과 같은 PCI 장치가 장착되며, PCI 버스에 연결되어 데이터의 송수신이 이루어진다.

【보정대상항목】 식별번호 50

【보정방법】 정정

【보정내용】

<50> 주문자의 주문에 따라 제작될 칩(300)은 일반적으로 도 2에 나타낸 바와 같은 복수개의 하드웨어 설계 블록들 및 소프트웨어 IP 설계 블록에 상응하는 하드웨어 IP 블럭들로 이루어진다.

【보정대상항목】 식별번호 51

【보정방법】 정정

【보정내용】

<51> 즉, 도면에 도시된 IP1, IP2(214)의 블록은 차후에 칩(300)의 설계 검증이 종료된 시점에서는 하드웨어 IP 블록으로 대치가 되며, 설계 검증 단계에서는 상응하는 소프트웨어 IP 설계 블록으로, 컴퓨터 시스템(2)의 시스템 메모리(130)의 일 영역내에 IP의 소프트웨어 모델로서(214) 상주한다.

【보정대상항목】 식별번호 52

【보정방법】 정정

【보정내용】

<52> 반면에, 오디오, 비디오, 인터페이스 블록, MCU의 블록은 하드웨어 설계 블록에 상응하는 개별소자 및 FPGA(Field Programmable Gate Array)로 구현될 수 있고, 메모리는 롬(Read Only Memory ; ROM) 및 램(Random Access Memory ; RAM)으로 구현되어 질 수 있다. 물론 이는 작업자 또는 사용자의 필요에 따라 다양한 방법으로 구현되어 질 수 있다.

【보정대상항목】 식별번호 54

【보정방법】 정정

【보정내용】

<54> 이와 같이 구성되는 칩은 도면에 도시된 바와 같이, 타겟(200)과 소프트웨어 IP(214)가 유기적으로 연결되어 사용자가 필요로 하는 동작을 수행하여 준다.

【보정대상항목】 식별번호 56

【보정방법】 정정

【보정내용】

<56> 컴퓨터 시스템(2)은 칩 설계 검증 프로그램(211), 시스템 메모리(130)상에 상주되는 칩 설계 검증용 메모리(212), 테스트 벤치(Test-bench)(213), 및 소프트웨어 IP(214)의 메모리 블록(210), 인터페이스 수단(220), 및 칩 설계 검증 프로그램(211)을 수행하고 제어하기 위한 CPU(미도시)를 구비한다.

【보정대상항목】 식별번호 61

【보정방법】 정정

【보정내용】

<61> 멀티 클럭 설정 값은 각 타겟 클럭의 초기 위치 지정 값과 기간, 하이레벨 점유시간 및 한 주기 점유 시간 값으로, 이러한 초기 위치 지정 값과 기간, 하이레벨 점유시간 및 한 주기 점유 시간 값은 인터페이스 수단(220)의 시스템 클럭 카운트 값으로 환산된 값이다.

【보정대상항목】 식별번호 62

【보정방법】 정정

【보정내용】

<62> 테스트 벤치(213)는 칩 설계 검증에 필요한 테스트 패턴들과 클럭 패턴을 가지는 멀티 클럭을 발생한다.

【보정대상항목】 식별번호 64

【보정방법】 정정

【보정내용】

<64> 인터페이스 수단(220)은 컴퓨터 시스템(2)의 확장 슬롯(170)에 설치되어 컴퓨터 시스템(2)의 외부에 존재하는 타겟(200)과의 연결을 도모하고, 타겟(200)의 출력데이터를 칩 설계 검증 프로그램(211)으로 전송하고, 칩 설계 검증 프로그램(211)을 통해 입력된 소프트웨어 IP(214)의 출력 데이터는 유효성 여부를 판단한 뒤, 소프트웨어 IP(214)의 유효한 출력데이터만을 타겟(200)으로 인가한다.

【보정대상항목】 식별번호 69

【보정방법】 정정

【보정내용】

<69> 타겟 입력용 레지스터(M1) 및 타겟 출력용 레지스터(M2) 영역의 행의 크기는 제 1 콘넥터(C1)를 통하여 타겟(200)에 연결되는 입출력 신호의 수에 대응되어 결정되며, 타겟 입력용 메모리(M3) 및 타겟 출력용 메모리(M4)의 크기는 PCI 버스(450)를 통하여 전송되는 소프트웨어 IP(214)와 타겟(200)간의 입출력 데이터 용량에 대응되어 결정된다.

【보정대상항목】 식별번호 71

【보정방법】 정정

【보정내용】

<71> 버스 스위치(420)는 제어부(430)와 타겟(200)간의 신호를 상호 연결하는 기능을 수행한다. 인터페이스 수단(220)에 입력되는 데이터에는 대부분의 경우, 타겟(200)으로 전송되어질 데이터 즉, 타겟의 입력 데이터와 타겟(200)으로부터 전송되는 데이터 즉, 타겟의 출력 데이터가 혼재되게 된다. 이에 버스 스위치(420)는 타겟의 입력 데이터와 타겟의 출력 데이터를 구분하여 타겟의 입력 데이터는 타겟의 입력용 메모리에 대응시키고, 타겟의 출력 데이터는 타겟의 출력용 메모리에 대응될 수 있도록 스위칭 한다.

【보정대상항목】 식별번호 76

【보정방법】 정정

【보정내용】

<76> 도 5는 도 4에 나타낸 제어부의 실시예의 블록도로서, 제어부(430)는 리셋 처리부(550), 플러그 & 플레이 제어부(552), PCI 로컬 버스 인터페이스 제어부(554), 어드레스 발생부(556), 메모리 제어부(558), 범용 레지스터 및 범용 레지스터 제어부(560), 인터럽트 제어부(562), 트리거 조건 제어부(564), 클록 제어부(566), 리드-백/JTAG 제어부(568), 병/직렬 변환부(570), 데이터 압축/복원 제어부(572), 타겟 인터페이스 제어부(574), 버스 스위치 및 종단 회로 제어부(576), 글리치(glitch) 검출부(578), 및 변화 발생 감지부(580)로 구성되어 있다.

【보정대상항목】 식별번호 83

【보정방법】 정정

【보정내용】

<83> 범용 레지스터 및 범용 레지스터 제어부(560)는 데이터 전송 동작에 따른 인터페이스 수단(220)의 각종 상태를 점검하고, 그 결과를 내부의 범용 레지스터(560)에 저장한다. 또한 멀티 클럭 설정 값을 저장한다.

【보정대상항목】 식별번호 86

【보정방법】 정정

【보정내용】

<86> 또한, 트리거 조건 제어부(564)는 장치 사용자가 GUI를 통하여 설정한 입력

또는 출력 프레임 스텝 수 또는 클럭 스텝 수를 트리거 조건으로 설정할 수도 있다. 입력 또는 출력 프레임 스텝 수가 트리거 조건 제어부(564)에 트리거 조건으로 설정된 경우에는 데이터 압축/복원 제어부(572)의 입/출력 처리 과정에서 카운팅되는 입력 또는 출력 프레임 수가 트리거 조건으로 설정된 입력 또는 출력 프레임 스텝 수와 일치되는 지점에서 트리거 동작을 수행하게 된다. 동일한 방법으로, 클럭 스텝 수가 트리거 조건으로 설정된 경우에는 클럭 제어부(568)로부터 발생되는 클럭 신호들 가운데 트리거 조건으로 설정된 클럭 신호의 카운팅 결과가 트리거 조건으로 설정된 클럭 스텝 수와 일치되는 지점에서 트리거 동작을 수행한다. 트리거 조건은 상술한 바와 같은 방법으로 개별적으로 적용하거나 우선순위를 두어서 혼용하여 적용할 수도 있다. 상술한 방법에 의해서 트리거 위치가 검출되면, 트리거 조건 제어부(564)는 장치 사용자에 의해서 설정된 GUI상의 디스플레이 기준 위치를 고려하여 어드레스 발생부(556)와 메모리 제어부(558)로부터 입력되는 트리거 지점에서의 어드레스와 메모리 영역 정보로부터 화면 디스플레이를 위한 시작 어드레스와 시작 어드레스의 메모리 영역, 종료 어드레스와 종료 어드레스의 메모리 영역에 관한 정보를 범용레지스터 제어부(560)에 저장하고, 인터럽트 제어부(562)에 디스플레이 해당 메모리 구간의 데이터 전송을 알리는 인터럽트 요청 신호를 전달한다. 또한, 장치 사용자에 의해서 지정된 GUI상의 표시 위치를 고려하여 어드레스 발생부(556)와 메모리 제어부(558)에 대한 기준 신호를 발생한다.

【보정대상항목】 식별번호 103

【보정방법】 정정

【보정내용】

<103> GUI상의 메뉴 항목을 이용하여 칩 설계 검증을 수행하고자 하는 소트웨어 IP(214) 및/또는 테스트 벤치를 선택하여 소트웨어 IP(214) 및/또는 테스트 벤치를 시스템 메모리(130)상에 상주시킨다(제 130 단계).

이하 소프트웨어 IP 및/또는 테스트 벤치로 표현되는 소프트웨어 블럭의 구성요소는 타겟(200)과의 상호 동작에 있어 올바게 표현하여야 할 때를 제외하고 설명의 편의를 위하여 간단히 " 소프트웨어 IP" 로 총칭한다.

【보정대상항목】 식별번호 109

【보정방법】 정정

【보정내용】

<109> 제 180 단계에서, GUI상의 메뉴 항목을 이용하여, 장치 사용자는 타겟(200)의 동작 모드를 설정하여 줄 수 있다.

【보정대상항목】 식별번호 111

【보정방법】 정정

【보정내용】

<111> GUI상의 메뉴 항목 또는 별도의 외부 입력 장치(미도시)를 이용하거나 혹은 테스트 벤치에 의하여 소프트웨어 IP(214)를 포함한 타겟(200)이 초기화 동작을 요청받으면, 칩 설계 검증 프로그램(211)은 타겟의 초기화 신호를 인터페이스 수단(220)으로 전송(A)한다(제 190 단계).

【보정대상항목】 식별번호 112

【보정방법】 정정

【보정내용】

<112> 도 7a는 도 6의 멀티 클럭 및 시스템 클럭의 설정 과정을 상세히 설명하기 위한 흐름도이고, 도 7b는 도 7a의 멀티 클럭 및 시스템 클럭의 다이아 그램을 나타내는 도면이다.

【보정대상항목】 식별번호 121

【보정방법】 정정

【보정내용】

<121> 또한 제 151 단계에서 테스트 벤치를 이용하여 멀티 클럭을 정의하는 경우, 칩 설계 검증 프로그램(211)은 테스트 벤치(213)의 클럭 정의 부분을 탐색하여, 칩 설계 검증을 위한 멀티 클럭을 구성하는 클럭들의 종류 및 각 클럭의 동작 속도를 획득한다.

【보정대상항목】 식별번호 122

【보정방법】 삭제

【보정대상항목】 식별번호 126

【보정방법】 정정

【보정내용】

<126> 장치 사용자는 GUI 상의 메뉴항목을 이용하여 디스플레이된 타겟 클럭들에 대하여 클럭 상호간의 초기 위치 지정(P1, P2, Pn)과 클럭 듀티(duty)(D1, D2, Dn)와 같은 타겟 클럭의 속성을 지정한다(제 157 단계).

【보정대상항목】 식별번호 127

【보정방법】 정정

【보정내용】

<127> 각 타겟 클럭의 초기 위치 지정(P1, P2, Pn) 값과 기간, 하이레벨 점유 기간 및 한주기 점유기간을 시스템 클럭카운트 값으로 환산하여 멀티 클럭 설정 값을 획득하고, 이를 인터페이스 수단(220)에 전송한다(제 159 단계).

【보정대상항목】 식별번호 128

【보정방법】 정정

【보정내용】

<128> 제 159 단계에서 장치 사용자는 인터페이스 수단(220)의 시스템 클럭의 속도를 선택하고, 선택된 속도에 대응되는 속도를 가지는 시스템 클럭을 도 7b의 (b)에 도시된 바와 같이 칩 설계 검증 프로그램(211)의 클럭 표시창에 디스플레이한다.

【보정대상항목】 식별번호 129

【보정방법】 정정

【보정내용】

<129> 칩 설계 검증 프로그램(211)은 멀티 클럭을 구성하는 각 타겟 클럭의 초기 위치 지정(P1, P2, Pn) 값과 기간, 하이레벨 점유 기간 및 한주기 점유기간을 시스템 클럭카운트 값으로 환산하여 멀티 클럭 설정 값을 획득하고, 이를 컴퓨터 시스템(2)의 시스템 메모리(130)의 칩 설계 검증용 메모리(212)와 인터페이스 수단(220)의 제어부(430)의 범용 레지스터(560)에 저장한다.

【보정대상항목】 식별번호 133

【보정방법】 정정

【보정내용】

<133> 이에 본 발명에서는 종래의 기술에서 발생하는 인터페이스 수단(200)의 동작 지연에 의한 데이터의 전송 지연 즉, 컴퓨터 시스템(2)내에서 존재하는 테스트 벤치(213)로부터 타겟(200) 인가용 멀티 클럭을 제공받음으로써 발생하는 전송 지연을 획기적으로 감소시켜 줄 수 있게 된다.

【보정대상항목】 식별번호 134

【보정방법】 정정

【보정내용】

<134> 도 8과 도 9a 내지 9b는 본 발명의 실시 예에 따른 칩 설계 검증 방법을 설명하기 위한 흐름도로, 도 8은 본 발명의 실시예에 따른 칩 설계 검증 방법의 초기화 동작을 설명하기 위한 흐름도이고, 도 9a는 본 발명의 실시예에 따른 칩 설계

검증 방법에 있어서 인터페이스 수단의 타겟에 대한(이하에서는 타겟측이라 지칭한다) 메인 동작을 설명하기 위한 흐름도이고, 도 9b는 본 발명의 실시예에 따른 칩 설계 검증 방법에 있어서 칩 설계 검증 프로그램의 소프트웨어 IP에 대한(이하에서는 IP측이라 지칭한다) 메인 동작을 설명하기 위한 흐름도이다.

【보정대상항목】 식별번호 137

【보정방법】 정정

【보정내용】

<137> 칩 설계 검증 프로그램(211)의 동작 개시와 더불어 소프트웨어 블록 또는 타겟의 구성요소의 여하에 따라서 인터페이스 수단(220)이 외부로부터 초기화 입력 신호를 수신하거나, GUI 메뉴항목의 설정에 의한 초기화 입력 신호를 수신하거나, 또는 테스트 벤치로부터 출력되는 초기화 입력 신호를 수신하게 되면(제 210 단계), 인터페이스 수단(220)은 수신된 초기화 신호를 타겟(200)측으로 전송한다.

【보정대상항목】 식별번호 139

【보정방법】 정정

【보정내용】

<139> 인터페이스 수단(220)은 초기화된 타겟(200)의 출력값 발생을 감지함에 따라, 소프트웨어 IP의 초기화 신호를 생성하고, "타겟 클럭 카운트 값"을 0으로 설정한다. 그리고 생성된 소프트웨어 IP의 초기화 신호와, 0으로 설정된 "타겟 클럭 카운트 값" 및 발생된 초기화된 타겟(200)의 출력값으로 구성된 타겟의 출력 데이터

를 IP측으로 전송한다(제 230 단계).

【보정대상항목】 식별번호 141

【보정방법】 정정

【보정내용】

<141> 그리고 생성된 출력 데이터를 인터페이스 수단(220)의 메모리(400)의 타겟 출력용 메모리(M4)의 영역에 저장한 후, 인터럽트 제어부(562)를 통해 전송 요청 인터럽트 신호를 발생하여 IP측으로 전송한다.

【보정대상항목】 식별번호 144

【보정방법】 정정

【보정내용】

<144> 칩 설계 검증 프로그램(211)은 제 230 단계에 의해 전송된 인터페이스 수단(220)의 초기화된 타겟(200)의 "타겟 클럭 카운트 값"과 "타겟의 출력값"으로 구성되는 출력 데이터와 소프트웨어 IP의 초기화 신호를 수신하면(제 260 단계), 수신된 타겟(200)의 출력값과 소프트웨어 IP의 초기화 신호를 소프트웨어 IP(214)에 입력한다.

【보정대상항목】 식별번호 147

【보정방법】 정정

【보정내용】

<147> 칩 설계 검증 프로그램(211)은 "IP 클럭 카운트 값"을 0으로 설정하고, 0으로 설정된 "IP 클럭 카운트 값" 및 발생된 초기화된 "IP의 출력 값"을 타겟측으로 전송한다(제 280 단계).

【보정대상항목】 식별번호 148

【보정방법】 정정

【보정내용】

<148> 이어서, 전송한 초기화된 "IP의 출력 값"에 대한 "ACKN 신호"가 타겟측으로부터 전송되는 지를 모니터링 하여 대기하고, 마침내 타겟측으로부터 "ACKN 신호"가 수신되면(제 290 단계), 초기화 동작을 정상적으로 수행하였음을 확인하고 IP측의 메인 동작으로 진입(D)한다.

【보정대상항목】 식별번호 152

【보정방법】 정정

【보정내용】

<152> 제 390 단계의 모니터링 결과, "타겟의 출력값"이 변화되지 않았으면, "IP 클럭 카운트 값과 비교 플래그"가 설정된 상태 인지를 확인하는 단계(제 400 단계)를 거쳐 다시 제 370 단계로 진행한다.

【보정대상항목】 식별번호 153

【보정방법】 정정

【보정내용】

<153> 인터페이스 수단(220)은 제 370 단계에서 제 400단계 사이를 반복 수행하면서 "타겟의 출력값"에 변화가 발생되었는지를 모니터링하면서 IP측으로부터의 진입(G)을 대기한다.

【보정대상항목】 식별번호 155

【보정방법】 정정

【보정내용】

<155> 제 320 단계의 확인 결과, "IP 클럭 카운트 값"이 "타겟 클럭 카운트 값"보다 작거나 같으면, IP측으로 “타겟 클럭 카운트 값과 비교 플래그 해제” 와 “ACKN 신호” 를 전송하고(제 340 단계), 인터페이스 수단(220)은 "IP의 출력값"을 타겟(200)의 입력 데이터로 갱신한다(제 345 단계).

【보정대상항목】 식별번호 160

【보정방법】 정정

【보정내용】

<160> 제 410 단계의 확인결과, "타겟 클럭 카운트 값"이 "IP 클럭 카운트 값"보다 작거나 같으면, 다시 제 370 단계에서 제 410 단계 사이를 반복 수행하면서 "타겟의 출력값"에 변화가 발생되는지를 모니터링한다.

【보정대상항목】 식별번호 162

【보정방법】 정정

【보정내용】

<162> 그리고 인터페이스 수단(220)은 IP측으로부터 전송한 데이터에 대한 "ACKN 신호"를 수신되는지를 모니터링 하면서 대기하고, IP측으로부터 "ACKN 신호"가 수신되면(제 440 단계), 곧바로 제 345 단계로 진입(F)을 수행한다.

【보정대상항목】 식별번호 163

【보정방법】 정정

【보정내용】

<163> 반면에 제 410 단계의 확인결과, "타겟 클럭 카운트 값"이 "IP 클럭 카운트 값"보다 클 때까지 "타겟의 출력값"이 변화되지 않으면, 인터페이스 수단(200)은 "IP 클럭 카운트 값과 비교 플래그"를 해제하고(제 420 단계), IP 측으로 "ACKN 신호" 를 전송한 후에(제 340 단계), 전송된 소프트웨어 IP(214)의 출력 데이터를 새로운 입력 데이터로 갱신하는 단계(제 345 단계)로 진행한다.

【보정대상항목】 식별번호 169

【보정방법】 정정

【보정내용】

<169> 제 520 단계의 확인 결과,"타겟 클럭 카운트 값"이 "IP 클럭 카운트 값" 보다 작거나 같으면, 타겟측으로 "IP 클럭 카운트 값과 비교 플래그 해제" 및 " ACKN 신호" 를 전송한다(제 540 단계).

【보정대상항목】 식별번호 175

【보정방법】 정정

【보정내용】

<175> 제 610 단계의 확인결과, "IP 클럭 카운트 값"이 "타겟 클럭 카운트 값"보다 작거나 같으면, 다시 제 570 단계에서 제 610 단계 사이를 반복 수행하면서 "소프트웨어 IP의 출력값"에 변화가 발생되는 지를 모니터링한다.

【보정대상항목】 식별번호 179

【보정방법】 정정

【보정내용】

<179> 그리고 칩 설계 검증 프로그램(211)은 타겟측으로부터 전송한 데이터에 대한 "ACKN 신호"가 수신되는 지를 모니터링 하면서 대기하고, 타겟측으로부터 "ACKN 신호"가 전달되면(제 640 단계), 누적된 "IP 클럭 카운트 값"에 따라 변화된 "소프트웨어 IP의 출력값"을 파형 관측창에 디스플레이 한다. 즉, 파형 관측창에 소프트웨어 IP(214)의 출력 파형을 갱신하고(제 645 단계), 제 542 단계로 진입(H)을 수행한다.

【보정대상항목】 식별번호 181

【보정방법】 정정

【보정내용】

<181> 또한 제 610 단계의 확인결과, "IP 클럭 카운트 값"이 "타겟 클럭 카운트 값"보다 클 때까지 "IP의 출력값"에 변화가 발생하지 않으면, 칩 설계 검증 프로그

램(211)은 "타겟 클럭 카운트 값과 비교 플래그"를 해제하고(제 620 단계), 타겟측으로 "ACKN 신호"를 전송한 후에(제 450 단계), 전송된 타겟(200)의 출력 데이터를 새로운 입력 데이터로 갱신하는 단계(제 542 단계)로 진행한다.

【보정대상항목】 식별번호 185

【보정방법】 정정

【보정내용】

<185> 따라서 본 발명의 칩 설계 검증 장치 및 방법은 칩 설계 검증 프로그램 및 인터페이스 수단을 이용하여 칩 설계 단계에서 칩 설계의 오류 및 에러를 검증할 수 있도록 하는 칩의 설계의 정확성을 제공하고, 칩 설계의 소요 시간 및 비용을 감소시켜 준다.

【보정대상항목】 식별번호 186

【보정방법】 정정

【보정내용】

<186> 또한 칩 설계 검증 장치 및 방법은 소프트웨어 IP 또는 타겟의 출력 데이터가 변화되었을 경우에만, 즉 이벤트가 발생한 경우에만 상대측과 데이터 송수신을 수행하도록 하여 데이터 전송 속도 및 효율을 증대한다.

【보정대상항목】 식별번호 187

【보정방법】 정정

【보정내용】

<187> 또한 칩 설계 검증 장치 및 방법은 데이터 전송 시, 소프트웨어 IP 또는 타겟의 변화된 출력 데이터를 추출하여 압축하고, 이를 상대측에 전송하도록 하여 데이터 전송 속도 및 효율을 증대한다.

【보정대상항목】 청구항 1

【보정방법】 정정

【보정내용】

【청구항 1】

적어도 하나 이상의 하드웨어 블록; 및

상기 하드웨어 블록과 데이터 통신을 수행하는 적어도 하나 이상의 소프트웨어 블록을 포함하고, 상기 하드웨어 블록과 상기 소프트웨어 블록간의 동작을 검증하는 컴퓨터를 구비하고,

상기 컴퓨터는

상기 하드웨어 블록의 출력데이터를 전송하고, 상기 소프트웨어 블록의 출력 데이터의 유효성 여부를 판단하고, 소프트웨어 블록의 유효한 출력데이터만을 상기 하드웨어 블록으로 인가하는 인터페이스 수단;

상기 소프트웨어 블록, 상기 하드웨어 블록과 상기 소프트웨어 블록간의 동작을 검증하는 칩 설계 검증 프로그램을 저장하는 저장수단; 및

상기 칩 설계 검증 프로그램을 수행하여 상기 소프트웨어 블록의 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 인터페이스 수단을 통해 입력되는 상

기 하드웨어 블록의 출력 데이터의 유효성 여부를 판단하고, 하드웨어 블록의 유효한 출력데이터만을 상기 소프트웨어 블록에 인가하도록 하는 제어부를 구비하는 것을 특징으로 하는 칩 설계 검증 장치.

【보정대상항목】 청구항 6

【보정방법】 정정

【보정내용】

【청구항 6】

제 5 항에 있어서, 상기 하드웨어 블록의 유효한 출력데이터는

상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같은 경우에는 상기 하드웨어 블록의 출력값이고, 상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커서 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않는 경우에는 상기 하드웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 장치.

【보정대상항목】 청구항 9

【보정방법】 정정

【보정내용】

【청구항 9】

제 8 항에 있어서, 상기 소프트웨어 블록의 유효한 출력데이터는

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 작거나 같은 경우에는 상기 소프트웨어 블록의 출력값이고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 커서 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 하드웨어 블록의 출력값이 변화하지 않는 경우에는 상기 소프트웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 장치.

【보정대상항목】 청구항 11

【보정방법】 정정

【보정내용】

【청구항 11】

적어도 하나 이상의 하드웨어 블록, 및 적어도 하나 이상의 소프트웨어 블록, 칩 설계 검증 프로그램, 및 입출력 데이터를 저장하는 저장수단과 상기 소프트웨어 블록과 상기 하드웨어 블록의 인터페이스를 수행하는 인터페이스 수단을 구비하는 컴퓨터를 구비하는 칩 설계 검증 장치의 칩 설계 검증 방법에 있어서,

상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이

스 수단을 통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 하드웨어 블록의 유효한 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계; 및

상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 소프트웨어 블록의 유효한 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 14

【보정방법】 정정

【보정내용】

【청구항 14】

제 13 항에 있어서, 상기 하드웨어 블록의 유효한 출력 데이터는

상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같은 경우에는 상기 하드웨어 블록의 출력값이고, 상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커서 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않는 경우에는 상기 하드웨어

블록의 출력값인 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 16

【보정방법】 정정

【보정내용】

【청구항 16】

제 15 항에 있어서, 상기 소프트웨어 블록의 유효한 출력 데이터는

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 작거나 같은 경우에는 상기 소프트웨어 블록의 출력값이고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 커서 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 하드웨어 블록의 출력값이 변화하지 않는 경우에는 상기 소프트웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 17

【보정방법】 정정

【보정내용】

【청구항 17】

제 16 항에 있어서, 상기 소프트웨어측 동작 단계는

상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통

해 상기 하드웨어의 출력 데이터를 수신하는 동작 진입 단계;

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 보다 작거나 같은 경우이거나, 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 보다 커서 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않는 경우이면, 상기 하드웨어 블록의 유효한 출력 데이터를 수신하였음을 확인하고 상기 소프트웨어 블록에 입력하는 입력 단계;

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 수신한 상기 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시키기 이전에 상기 소프트웨어 블록의 출력값이 변화하면 상기 소프트웨어 블록의 출력 데이터를 상기 인터페이스 수단으로 전송하는 전송 단계; 및

상기 입력 단계 또는 상기 전송 단계가 완료되면, 상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 초기화하고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 모니터링하는 모니터링 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 18

【보정방법】 정정

【보정내용】

【청구항 18】

제 17 항에 있어서, 상기 입력 단계는

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같으면, 상기 수신한 하드웨어 블록의 출력값을 상기 하드웨어 블록의 유효한 출력 데이터로서 상기 소프트웨어 블록에 입력하는 단계;

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 크면, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값보다 커질 때까지 상기 소프트웨어 블록의 출력값이 변화되지 않으면, 상기 수신한 하드웨어 블록의 출력값을 상기 하드웨어 블록의 유효한 출력 데이터로서 상기 소프트웨어 블록에 입력하는 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 19

【보정방법】 정정

【보정내용】

【청구항 19】

제 18 항에 있어서, 상기 입력 단계는

상기 칩 설계 검증 프로그램을 수행하여 상기 컴퓨터의 화면에 상기 하드웨어 블록의 유효한 출력 데이터를 디스플레이하는 단계를 더 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 21

【보정방법】 정정

【보정내용】

【청구항 21】

제 20 항에 있어서, 상기 전송 단계는

상기 칩 설계 검증 프로그램을 수행하여 상기 컴퓨터의 화면에 상기 소프트웨어 블록의 유효한 출력 데이터를 디스플레이하는 단계를 더 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 23

【보정방법】 정정

【보정내용】

【청구항 23】

제 16 항에 있어서, 상기 하드웨어측 동작 단계는

상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 발생되는 상기 소프트웨어 블럭의 출력 데이터를 수신하는 동작 진입 단계;

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 작거나 같은 경우이거나, 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 커서 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 인터페이스 수단의 출력값이 변화하지 않는 경우이면, 상기 소프트웨어 블록의 유효한 출력 데이터를 수신하였음을 확인하고 상기 하드웨어 블록에 입력하는 입력 단계;

상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시키기 이전에 상기 하드웨어 블록의 출력값이 변화하면 상기 하드웨어 블록의 출력 데이터를 상기 칩 설계 검증 프로그램으로 전송하는 전송 단계; 및

상기 입력 단계 또는 상기 전송 단계가 완료되면, 상기 증가된 상기 인터페이스 수단의 시스템 클럭 카운트 값을 초기화하고, 상기 인터페이스 수단의 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 모니

터링하는 모니터링 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 24

【보정방법】 정정

【보정내용】

【청구항 24】

제 23 항에 있어서, 상기 입력 단계는

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 작거나 같으면, 상기 수신한 소프트웨어 블록의 출력값을 상기 소프트웨어 블록의 유효한 출력 데이터로서 상기 하드웨어 블록에 입력하는 단계;

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 크면, 상기 인터페이스 수단의 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 확인하는 단계;

상기 증가된 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커질 때까지 상기 하드웨어 블록의 출력값이 변화되지 않으면, 상기 수신한 소프트웨어 블록의 출력값을 상기 소프트웨어 블록의 유효한 출력 데이터로서 상기 하드웨어 블록에 입력하는 입력 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 25

【보정방법】 정정

【보정내용】

【청구항 25】

제 24 항에 있어서, 상기 전송 단계는

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 크면, 상기 인터페이스 수단이 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같은 경우에 상기 하드웨어 블록의 출력값이 변화되면, 상기 하드웨어의 출력 데이터를 상기 인터페이스 수단에 전송하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 26

【보정방법】 정정

【보정내용】

【청구항 26】

적어도 하나 이상의 기능 블록을 구비하는 소프트웨어 블록 및 하드웨어 블록, 상기 소프트웨어 블록과 상기 하드웨어 블록의 동작을 검증하기 위한 칩 설계

검증 프로그램, 상기 칩 설계 검증 프로그램을 수행함에 의해 발생되는 상기 소프트웨어 블록의 입출력 데이터를 저장하는 저장부, 및 상기 하드웨어 블록과 상기 소프트웨어 블록간의 인터페이스를 수행하는 인터페이스 수단을 구비하는 칩 설계 검증 장치의 칩 설계 검증 방법에 있어서,

상기 칩 설계 검증 프로그램은 멀티 클럭 설정값을 획득하고, 상기 인터페이스 수단에 제공하고, 상기 멀티 클럭 설정값과 상기 칩 설계 검증 프로그램의 시스템 클럭에 따라 멀티 클럭을 발생하여 상기 소프트웨어 블록에 인가하고, 상기 인터페이스 수단은 상기 멀티 클럭 설정값과 상기 인터페이스 수단의 시스템 클럭에 따라 멀티 클럭을 발생하여 상기 하드웨어 블록에 인가하는 클럭 생성 단계;

상기 칩 설계 검증 프로그램의 멀티 클럭에 응답하여 동작하는 상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 하드웨어 블록의 유효한 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계; 및

상기 인터페이스 수단의 멀티 클럭에 응답하여 동작하는 상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 소프트웨어 블록의 유효한 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계

를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 28

【보정방법】 정정

【보정내용】

【청구항 28】

제 27 항에 있어서, 상기 하드웨어 블록의 유효한 출력 데이터는

상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같은 경우에는 상기 하드웨어 블록의 출력값이고, 상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커서 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않는 경우에는 상기 하드웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 30

【보정방법】 정정

【보정내용】

【청구항 30】

제 29 항에 있어서, 상기 소프트웨어 블록의 유효한 출력 데이터는

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수

단의 시스템 클럭 카운트 값보다 작거나 같은 경우에는 상기 소프트웨어 블록의 출력값이고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 커서 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 하드웨어 블록의 출력값이 변화하지 않는 경우에는 상기 소프트웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 31

【보정방법】 정정

【보정내용】

【청구항 31】

제 30 항에 있어서, 상기 소프트웨어측 동작 단계는

상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 상기 하드웨어의 출력 데이터를 수신하는 동작 진입 단계;

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 보다 작거나 같은 경우이거나, 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 보다 커서 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않는 경우이면, 상기 하드웨어 블록

의 유효한 출력 데이터를 수신하였음을 확인하고 상기 소프트웨어 블록에 입력하는 입력 단계;

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 수신한 상기 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시키기 이전에 상기 소프트웨어 블록의 출력값이 변화하면 상기 소프트웨어 블록의 출력 데이터를 상기 인터페이스 수단으로 전송하는 전송 단계; 및

상기 입력 단계 또는 상기 전송 단계가 완료되면, 상기 증가된 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 초기화하고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 모니터링하는 모니터링 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 32

【보정방법】 정정

【보정내용】

【청구항 32】

제 31 항에 있어서, 상기 입력 단계는

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같으면, 상기 수신한 하드웨어 블록의 출력값을 상기 하드웨어 블록의 유효한 출력 데이터로서 상기 소프트웨어 블록

에 입력하는 단계;

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 크면, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값보다 커질때까지 상기 소프트웨어 블록의 출력값이 변화되지 않으면, 상기 수신한 하드웨어 블록의 출력값을 상기 하드웨어 블록의 유효한 출력 데이터로서 상기 소프트웨어 블록에 입력하는 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 34

【보정방법】 정정

【보정내용】

【청구항 34】

제 31 항에 있어서, 상기 전송 단계는

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 크면, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값보다 작거나 같은 경우에 상기 소프트웨어 블록의 출력값이 변화되면, 상기 소프트웨어 블럭의 출력 데이터를 상기 인터페이스 수단에 전송하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 35

【보정방법】 정정

【보정내용】

【청구항 35】

제 34 항에 있어서, 상기 전송 단계는

상기 칩 설계 검증 프로그램을 수행하여 상기 컴퓨터의 화면에 상기 소프트웨어 블록의 유효한 출력 데이터를 디스플레이하는 단계를 더 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 37

【보정방법】 정정

【보정내용】

【청구항 37】

제 30 항에 있어서, 상기 하드웨어측 동작 단계는

상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 발생되는 상기 소프트웨어 블록의 출력 데이터를 수신하는 동작 진입 단계;

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값이 보다 작거나 같은 경우이거나, 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값이 보다 커서 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 인터페이스 수단의 출력값이 변화하지 않는 경우이면, 상기 소프트웨어 블록의 유효한 출력 데이터를 수신하였음을 확인하고 상기 하드웨어 블록에 입력하는 입력 단계;

상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시키기 이전에 상기 하드웨어 블록의 출력값이 변화하면 상기 하드웨어 블록의 출력 데이터를 상기 칩 설계 검증 프로그램으로 전송하는 전송 단계; 및

상기 입력 단계 또는 상기 전송 단계가 완료되면, 상기 증가된 인터페이스 수단의 시스템 클럭 카운트 값을 초기화하고, 상기 인터페이스 수단의 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 모니터링하는 모니터링 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 38

【보정방법】 정정

【보정내용】

【청구항 38】

제 37 항에 있어서, 상기 입력 단계는

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 작거나 같으면, 상기 수신한 소프트웨어 블록의 출력값을 상기 소프트웨어 블록의 유효한 출력 데이터로서 상기 하드웨어 블록에 입력하는 단계;

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 크면, 상기 인터페이스 수단이 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커질 때까지 상기 하드웨어 블록의 출력값이 변화되지 않으면, 상기 수신한 소프트웨어 블록의 출력값을 상기 소프트웨어 블록의 유효한 출력 데이터로서 상기 하드웨어 블록에 입력하는 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 청구항 39

【보정방법】 정정

【보정내용】

【청구항 39】

제 37 항에 있어서, 상기 전송 단계는

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 크면, 상기 인터페이스 수단의 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

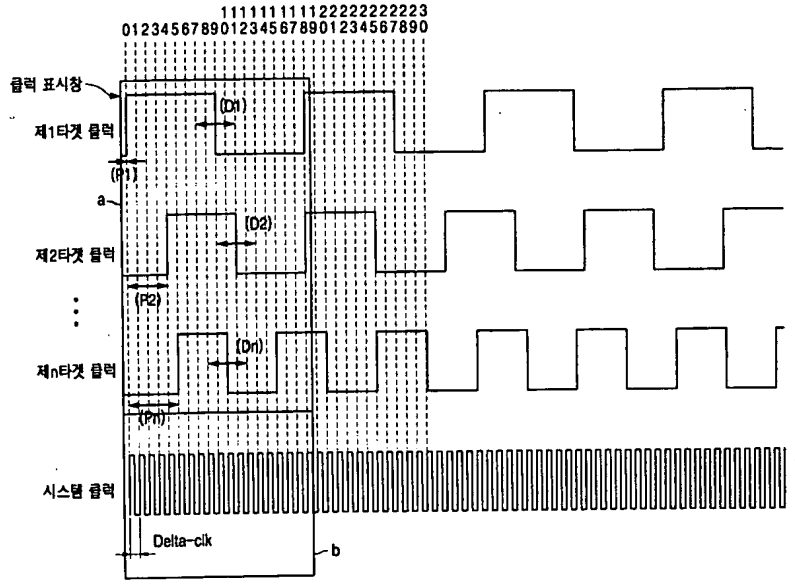
상기 증가된 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같은 경우에 상기 하드웨어 블록의 출력값이 변화되면, 상기 하드웨어의 출력 데이터를 상기 칩 설계 검증 프로그램으로 전송하는 것을 특징으로 하는 칩 설계 검증 방법.

【보정대상항목】 도 7b

【보정방법】 정정

【보정내용】

【도 7b】

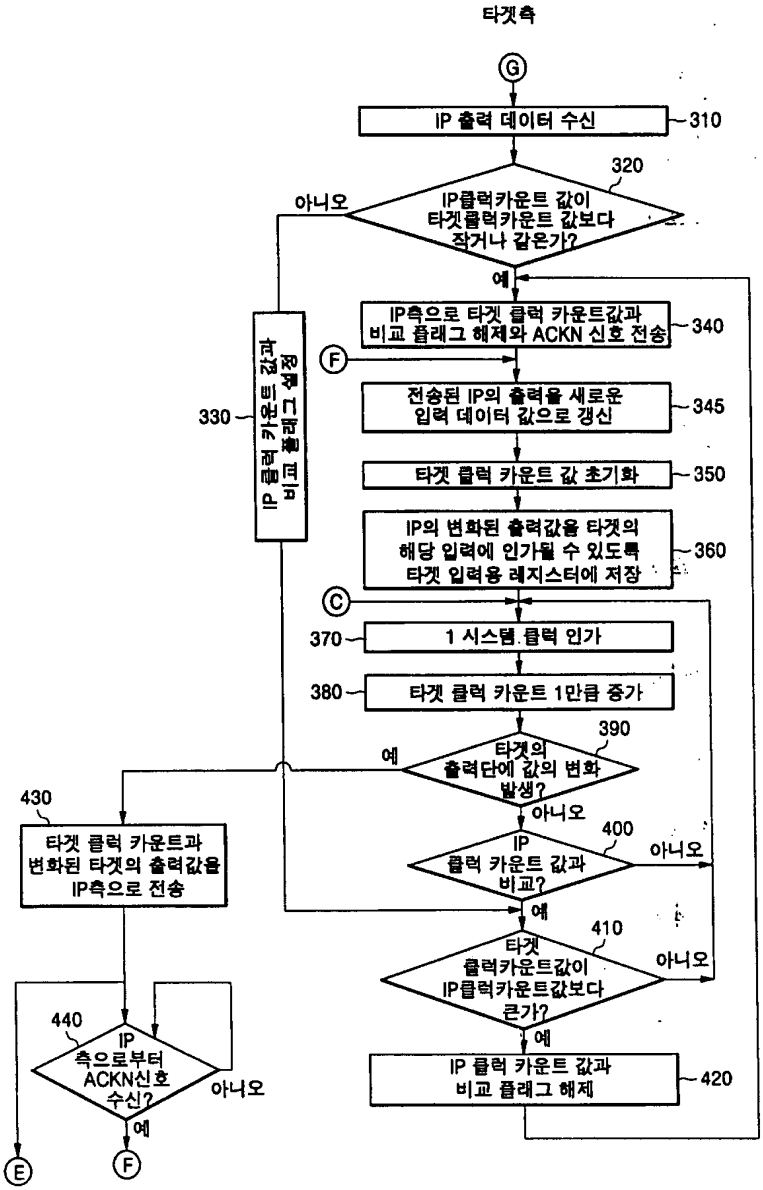


【보정대상항목】 도 9a

【보정방법】 정정

【보정내용】

【도 9a】

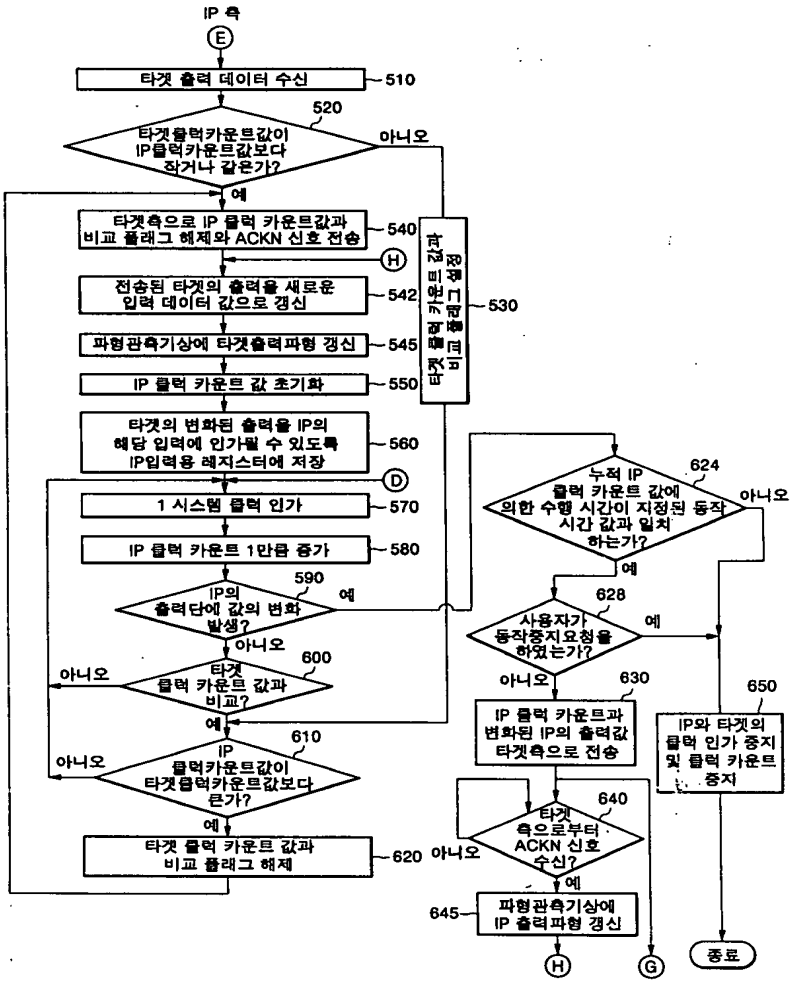


【보정대상항목】 도 9b

【보정방법】 정정

【보정내용】

【도 9b】


IP 측
E
타겟 출력 데이터 수신
510
520
타겟클럭카운트값이 IP클럭카운트값보다 작거나 같은가?
아니오
예
타겟측으로 IP 클럭 카운트값과 비교 플래그 해제와 ACKN 신호 전송
540
H
전송된 타겟의 출력을 새로운 입력 데이터 값으로 갱신
542
파형관측기상에 타겟출력파형 갱신
545
IP 클럭 카운트 값 초기화
550
타겟의 변화된 출력을 IP의 해당 입력에 인가될 수 있도록 IP입력용 레지스터에 저장
560
D
1 시스템 클럭 인가
570
IP 클럭 카운트 1만큼 증가
580
590
IP의 출력단에 값의 변화 발생?
예
아니오
600
타겟 클럭 카운트 값과 비교?
아니오
예
610
IP 클럭카운트값이 타겟클럭카운트값보다 큰가?
아니오
예
타겟 클럭 카운트 값과 비교 플래그 해제
620
타겟 클럭 카운트 값과 비교 플래그 설정
530
624
누적 IP 클럭 카운트 값에 의한 수행 시간이 지정된 동작 시간 값과 일치 하는가?
아니오
예
628
사용자가 동작중지요청을 하였는가?
예
아니오
630
IP 클럭 카운트과 변화된 IP의 출력값 타겟측으로 전송
640
타겟 측으로부터 ACKN 신호 수신?
아니오
예
645
파형관측기상에 IP 출력파형 갱신
650
IP와 타겟의 클럭 인가 중지 및 클럭 카운트 중지
종료
H
G

【서지사항】

【서류명】 특허출원서

【권리구분】 특허

【수신처】 특허청장

【제출일자】 2004.02.17

【발명의 국문명칭】 칩 설계 검증 장치 및 방법

【발명의 영문명칭】 chip design verification apparatus and method

【출원인】

【성명】 박현주

【출원인코드】 4-2000-026787-4

【대리인】

【성명】 박상수

【대리인코드】 9-1998-000642-5

【포괄위임등록번호】 2000-053697-9

【발명자】

【성명】 박현주

【출원인코드】 4-2000-026787-4

【심사청구】 청구

【취지】 특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다.

대리인 박상수 (인)

【수수료】

【기본출원료】	65 면	38,000 원
【가산출원료】	0 면	0 원
【우선권주장료】	0 건	0 원
【심사청구료】	39 항	1,357,000 원
【합계】	1,395,000 원	

【감면사유】 개인(70%감면)

【감면후 수수료】 418,500 원

【요약서】

【요약】

본 발명은 칩 설계 검증 장치 및 방법을 공개한다. 그 방법은 상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 유효한 하드웨어 블록의 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계와, 상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 유효한 소프트웨어 블록의 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계를 구비하는 것을 특징으로 한다. 따라서 칩 설계시에 칩 설계상의 오류를 검출할 수 있도록 하고, 보다 빠른 칩 설계 검증 속도를 제공하여 준다.

【대표도】

도 9a

【명세서】

【발명의 명칭】

칩 설계 검증 장치 및 방법{chip design verification apparatus and method}

【도면의 간단한 설명】

<1> 도 1은 일반적인 컴퓨터 시스템의 블록도이다.

<2> 도 2는 본 발명의 칩 설계 검증 장치의 블록도이다.

<3> 도 3은 일반적인 소프트웨어 IP 및 타겟을 구비하는 칩의 개념도이다.

<4> 도 4는 본 발명의 실시예에 따른 인터페이스 수단의 블록도이다

<5> 도 5는 도4에 나타낸 제어부의 실시예의 블록도이다.

<6> 도 6은 본 발명의 칩 설계 검증 방법을 수행함에 따른 컴퓨터 시스템 내부의 동작으로 설명하기 위한 흐름도이다.

<7> 도 7a 및 도 7b는 도 6의 멀티 클럭 및 기준 클럭의 발생 과정을 상세히 설명하기 위한 흐름도이다.

<8> 도 8은 본 발명의 칩 설계 검증 방법을 수행함에 따른 초기화 동작을 설명하기 위한 흐름도이다.

<9> 도 9a 및 도 9b는 본 발명의 칩 설계 검증 방법을 수행함에 따른 메인 동작을 설명하기 위한 흐름도이다.

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<10> 본 발명은 칩 설계 검증 장치에 관한 것으로, 특히 소프트웨어 IP를 활용한 칩의 설계 검증 장치 및 방법에 관한 것이다.

<11> 오늘날 반도체 설계 자동화와 관련된 각종 EDA(Electronic Design Automation) 도구(tool)의 보급이 확대되고 HDL(Hardware Description Language)를 이용한 설계 방법이 보편화되면서 ASIC(Application Specific Integrated Circuit)의 설계 환경은 크게 개선되었다.

<12> 따라서 설계되는 회로의 크기도 기존의 수만~수십만 게이트 수준에서 근래에는 수백만~수천만 게이트에 이르는 정도의 용량을 설계하기에 이르러 ASIC은 하나의 칩에 시스템을 구현하는 것이 가능한 정도에 이르렀다. 이와 같이 구성된 시스템을 SoC(System on a Chip)이라고 한다.

<13> 또한 ASIC 또는 SoC를 설계할 경우, 모든 것을 제로베이스에서 개발하기 보다는 기존에 이미 존재하는 설계 회로의 일부 기능 블록을 변경하거나 추가하는 형태로 진행이 되는 경우가 보다 일반적인 경우이며, 변경되거나 추가되는 기능 블록 또한 기존에 단위 기능 블록 형태로 존재하는 IP(Intellectual Property) 블록을 활용하는 것이 근래의 ASIC 또는 SoC 설계의 추세라고 할 수 있다.

<14> 이와 같은 다양한 어플리케이션들을 제공하며, 기존에 미리 설계된 모델들로 제공되는 IP는 크게 소프트웨어 IP와 하드웨어 IP로 나뉘어 진다.

<15> 소프트웨어 IP는 원시 코드의 코딩 형태에 따라, C, C++, System C, HDL(Hardware Description Language)(VHDL, Verilog HDL, System Verilog, ect)등의 언어로 구현되는 IP이고, 하드웨어 IP는 존재하는 형태에 따라서 주문형 반도체(ASIC), 프로그래머블반도체(FPGA) 또는 이들을 포함한 여러 디바이스를 탑재한 보드등의 형태로 구현되는 IP이다.

<16> 이때 IP를 하드웨어 모델로 거래 또는 배포하는 경우에는 여러 가지 측면에서 불편한 점이 있는데, 일예로는 다음과 같은 것들이 있다.

<17> 첫째, 샘플 형태로 전달될 하드웨어 IP가 볼그레이드어레이(BGA)ㅇ 또는 다수 핀의 ASIC 또는 FPGA 형태인 경우, 사용을 위해서는 하드웨어 IP가 탑재될 새로운 보드를 제작하여야 하고 보드 제작 이후에는 조립 상태의 여하에 따라서는 IP 동작이 불량 현상으로 나타나는 경우가 빈발하다.

<18> 둘째, 하드웨어 IP가 FPGA 형태인 경우 SRAM 방식으로 사용되는 FPGA에 있어서는 별도의 프로그램 롬(PROM)이 함께 전달이 되어야 하는 데, 이 경우 PROM이 복제되어 도용될 문제점이 있다.

<19> 셋째, 하드웨어 IP는 물리적인 형상으로 구현되는 것으로, 샘플 배달에 따른 시간 지연이 필연적으로 발생하게 되는 등의 문제점이 있다.

<20> 반면에 IP를 소프트웨어 모델로 거래 또는 배포하는 경우에는 이상과 같은 문제점이 발생하지 않게 된다. 즉, IP의 사용시 신뢰성을 보장하면서도 IP의 보안상 취약점을 노출하지 않으며, 배달에 따른 시간 지연이 발생하지 않게 되는 장점

이 있는 반면에, 하드웨어 IP에 비하여 사용시 IP의 동작 속도가 현저히 느려지는 단점이 있다.

<21> 특히, 본 발명에서 관심을 가지는 분야는 이상과 같은 하드웨어 IP가 갖는 단점을 최소화하면서도 소프트웨어 IP의 본연의 장점을 제공하는 소프트웨어 IP를 활용하는 칩의 설계 검증 장치 및 방법에 관한 것이다.

<22> 칩의 설계 검증은 용인된 실체(accepted entity)에 대한 검사를 함축하는 것으로, 이는 칩 설계에 있어서 명세서를 배경으로 설계를 검사하는 것을 의미하며, 칩의 설계 검증의 목적은 실체의 한계 내에서 칩이 구현되고 제조된 후에 의도대로 작동할 것인지를 알아내는 데 있다.

<23> 따라서 칩이 소프트웨어 IP를 활용하는 경우, 상기의 칩 설계 검증의 목적을 구현하기 위해서는 칩의 하드웨어 설계 블록 즉 타겟 및 소프트웨어 IP의 공동 시뮬레이션을 수행하여 주어야 한다.

<24> 종래의 소프트웨어 IP를 이용하는 칩의 설계 검증 장치로는 미국 특허 번호 제6,356,862에 공개된 지연 동기를 이용한 하드웨어와 소프트웨어 상호 검증기가 있을 수 있겠다. 미국 특허 번호 제6,356,862에 공개된 장치는 하드웨어와 소프트웨어 상호 조정 수단을 구비하여 하드웨어 검증기와 소프트웨어 검증기가 상호 동기되어 검증 동작을 수행할 수 있도록 하였다.

<25> 이 장치는 하드웨어 검증기와 소프트웨어 검증기의 상호 동기를 위해 동기화 창을 이용하며, 이 동기화 창은 빠른 속도로 동작하는 검증 기에 상관없이 느린 속

도로 동작하는 검증기에 의해 크기가 정해진다.

<26> 이에 동작대기 시간이 증가되고, 이에 따라 전체적인 칩의 설계 검증 속도가 느려지게 되는 문제점이 있었다.

<27> 또한 이때의 소프트웨어 IP와 타겟의 칩 설계 검증에 필요한 테스트 케이스들과 클럭 패턴을 가지는 멀티 클럭을 제공하는 테스트 벤치는 컴퓨터 시스템 내에 존재하고, 타겟은 컴퓨터 시스템의 외부에 존재하여, 소프트웨어 IP와 타겟은 별도의 인터페이스 수단을 통해 데이터 통신을 수행하여야 했었다.

<28> 이에 칩의 설계 검증을 위한 공동 시뮬레이션(하드웨어/소프트웨어 co-simulation) 시에는 소프트웨어 IP와 타겟간의 데이터 통신의 효율 및 속도가 전반적인 성능에 있어서 중요한 문제로 부각되게 된다.

<29> 그러나 종래의 칩의 설계 검증 장치의 경우에는 일정 시간을 가지는 주기를 기반으로 하여 소프트웨어 IP 또는 타겟에서 발생된 출력 데이터를 상대측에 제공하였었다. 이에 데이터 통신 용량이 불필요하게 증가하게 되는 문제점이 발생하였었다.

<30> 또한 종래의 칩의 설계 검증 장치에서 타겟은 동작에 필요한 클럭(일반적으로 멀티 클럭)을 컴퓨터 시스템 내에 존재하는 테스트 벤치로부터 제공받아야 했다.

<31> 이에 인터페이스 수단의 동작 지연에 의한 데이터의 전송 지연이 필연적으로 발생하게 되어, 칩 설계 검증을 위한 타겟의 동작 속도가 불필요로 하게 느려지는

문제점이 있었다.

【발명이 이루고자 하는 기술적 과제】

<32> 발명의 목적은 칩 설계 단계에서 소프트웨어 IP와 타겟과의 상호 접속을 지원하여 칩 설계의 오류 및 에러를 검증할 수 있도록 하는 칩의 설계 검증 장치 및 방법을 제공하는데 있다.

<33> 본 발명의 다른 목적은 이벤트 기반으로 소프트웨어 IP 또는 타겟의 데이터 송수신이 수행되도록 하여 데이터 전송 속도 및 효율을 높이는 칩의 설계 검증 장치 및 방법을 제공하는데 있다.

<34> 본 발명의 또 다른 목적은 타겟의 동작을 위한 멀티 클럭을 인터페이스 수단에서 독립적으로 발생하도록 하여 타겟의 동작 속도를 증대하는 칩의 설계 검증 장치 및 방법을 제공하는데 있다.

<35> 상기 목적들을 달성하기 위한 본 발명의 칩 설계 검증 장치는 적어도 하나 이상의 하드웨어 블록과, 상기 하드웨어 블록과 데이터 통신을 수행하는 적어도 하나 이상의 소프트웨어 블록을 포함하고, 상기 하드웨어 블록과 상기 소프트웨어 블록간의 동작을 검증하는 컴퓨터를 구비하고, 상기 컴퓨터는 상기 하드웨어 블록의 출력데이터를 전송하고, 상기 소프트웨어 블록의 출력 데이터의 유효성 여부를 판단하고, 유효한 소프트웨어 블록의 출력데이터만을 상기 하드웨어 블록으로 인가하는 인터페이스 수단과, 상기 소프트웨어 블록, 상기 하드웨어 블록과 상기 소프트웨어 블록간의 동작을 검증하는 칩 설계 검증 프로그램을 저장하는 저장수단과, 상

기 칩 설계 검증 프로그램을 수행하여 상기 소프트웨어 블록의 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 인터페이스 수단을 통해 입력되는 상기 하드웨어 블록의 출력 데이터의 유효성 여부를 판단하고, 유효한 하드웨어 블록의 출력 데이터만을 상기 소프트웨어 블록에 인가하도록 하는 제어부를 구비하는 것을 특징으로 한다.

<36> 상기 목적들을 달성하기 위한 본 발명의 제 1 형태의 칩 설계 검증 방법은 상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 유효한 하드웨어 블록의 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계와, 상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 유효한 소프트웨어 블록의 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계를 구비하는 것을 특징으로 한다.

<37> 상기 목적들을 달성하기 위한 본 발명의 제 2 형태의 칩 설계 검증 방법은 상기 칩 설계 검증 프로그램은 멀티 클럭 설정값을 획득하고, 상기 인터페이스 수단에 제공하고, 상기 멀티 클럭 설정값과 상기 칩 설계 검증 프로그램의 시스템 클럭에 따라 멀티 클럭을 발생하여 상기 소프트웨어 블록에 인가하고, 상기 인터페이스 수단은 상기 멀티 클럭 설정값과 상기 인터페이스 수단의 시스템 클럭에 따라

멀티 클럭을 발생하여 상기 하드웨어 블록에 인가하는 클럭 생성 단계와, 상기 칩 설계 검증 프로그램의 멀티 클럭에 응답하여 동작하는 상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 유효한 상기 하드웨어 블록의 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계와, 상기 칩 설계 검증 프로그램의 멀티 클럭에 응답하여 동작하는 상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 유효한 상기 소프트웨어 블록의 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계를 구비하는 것을 특징으로 한다.

【발명의 구성】

<38> 이하, 첨부한 도면을 참고로 하여 본 발명의 칩 설계 검증 장치 및 방법을 설명하면 다음과 같다.

<39> 도 1은 일반적인 컴퓨터 시스템의 블록도로서, 모니터(1)와 본체(2)로 구성되고, 본체(2)는 CPU(Central Processing Unit) 또는 프로세서(Processor)(100), 그래픽 신호 처리 장치(110), 메모리 컨트롤러 허브(Memory Controller Hub ; MCH)(120), 시스템 메모리(130), 하드 디스크(140), 입출력 콘트롤러 허브(I/O Controller Hub; ICH)(160), 확장 슬롯들(170), ATA(Advanced Technology

Attachment)(150), 범용 직렬 버스(Universal Serial Bus ; USB)를 구비한다.

<40> 도 1에 나타낸 본체(2)의 블록들 각각의 구성을 설명하면 다음과 같다.

<41> CPU(100)는 응용 프로그램을 수행하고 전반적인 제어를 수행한다.

<42> 그래픽 신호 처리 장치(110)는 시스템 메모리(130)로부터 전송되는 그래픽 신호를 신호 처리하여 모니터(1)상에 디스플레이 한다.

<43> MCH(120)는 CPU(100)와 연결하는 호스트 인터페이스(Host Interface), 시스템 메모리(130)를 잇는 메모리 인터페이스(Memory Interface), AGP(Accelerated Graphics Port)버스와 연결해서 그래픽 신호 처리 장치(110)를 제어하는 AGP 인터페이스를 구비하여, CPU(100), 시스템 메모리(130), 그래픽 신호 처리 장치(110) 사이의 인터페이스를 수행한다.

<44> 시스템 메모리(130)는 CPU(100)가 보다 빠른 동작을 수행할 수 있도록 현재 실행 중인 응용프로그램 그리고 현재 사용 중인 데이터를 상주시키는 저장 수단으로 역할을 수행한다.

<45> 하드 디스크(140)에는 각종 응용 프로그램과 각종 데이터가 저장된다.

<46> ATA(150)은 MCH(120)와 하드 디스크(140)간의 인터페이스를 수행하고, USB (Universal Serial Bus)(180)는 직렬 포트의 일종으로 오디오플레이어, 조이스틱, 키보드, 전화, 스캐너 및 프린터 등과 같은 주변장치와 컴퓨터 시스템(2)간의 플러그 앤 플레이 인터페이스이다.

<47> ICH(160)는 확장 슬롯, ATA, USB 등 주변장치의 들의 전원 공급을 제어함과

동시에, 데이터 흐름을 제어하고, 확장 슬롯들, ATA, 및 USB과 MCH(120)간의 인터페이스를 수행한다.

<48> 확장 슬롯들(170)은 PCI(Peripheral Component Interconnect) 슬롯으로, LAN 카드, 팩스 모뎀 카드 등과 같은 PCI 장치가 장착되며, 확장 슬롯은 PCI 버스에 연결되어 데이터의 송수신이 이루어진다.

<49> 도 2는 일반적인 소프트웨어 IP 및 타겟을 구비하는 칩의 개념도를 나타낸 것이다.

<50> 주문자의 주문에 따라 제작될 칩(300)은 일반적으로 도 2에 나타낸 바와 같은 복수개의 하드웨어 설계 블록들 및 소프트웨어 설계 블록들로 이루어진다.

<51> 도면에 도시된 IP1, IP2(214)의 블록은 차후에 칩(300)의 설계 검증이 종료된 시점에서는 하드웨어 IP 블록으로 대치가 되며, 설계 검증 단계에서는 상응하는 소프트웨어 설계 블록으로, 컴퓨터 시스템(2)의 시스템 메모리(130)의 일영역내에 IP의 소프트웨어 모델로서(214)에 상주한다.

<52> 반면에, 오디오, 비디오, 인터페이스 블록, MCU의 블록은 하드웨어 설계 블록에 상응하는 개별소자 및 FPGA(Field Programmable Gate Array)로 구현될 수 있고, 메모리는 롬(Read Only Memory ; ROM) 및 램(Random Access Memory ; RAM)으로 구현되어 질 수 있다. 물론 이는 작업자 또는 주문자의 필요에 따라 다양한 방법으로 구현되어 질 수 있다.

<53> 본 발명에서는 오디오, 비디오, 인터페이스 블록, MCU등과 같은 칩(300)의

하드웨어 설계 블록들을 총칭하여 타겟(200)이라 지칭한다.

<54> 이와 같이 구성되는 칩은 도면에 도시된 바와 같이, 타겟(200)과 소프트웨어 IP(214)가 유기적으로 연결되어 주문자가 필요로 하는 동작을 수행하여 준다.

<55> 도 3은 본 발명의 실시예에 따른 칩 설계 검증 장치의 구성을 도식화한 블록도로서, 칩 설계 검증을 수행하는 컴퓨터 시스템의 본체(2)와 이에 연결되는 타겟(200)으로 구성된다.

<56> 컴퓨터 시스템(2)은 칩 설계 검증 프로그램(211), 시스템 메모리(130)상에 상주되는 칩 설계 검증용 메모리(212), 테스트 벤치(testbench)(213), 및 소프트웨어 IP(214)의 메모리 블록(210), 인터페이스 수단(220), 및 칩 설계 검증 프로그램(211)을 수행하고 제어하기 위한 CPU(미도시)를 구비한다.

<57> 타겟(200)은 이상에서 설명된 칩의 하드웨어 설계 블록으로서, 인터페이스 수단(220)을 통해 컴퓨터 시스템(2)의 본체에 연결된다.

<58> CPU(미도시)는 칩 설계 검증 프로그램(211)을 수행하여, 장치 사용자가 칩 설계 검증에 필요한 설정, 동작을 선택할 수 있도록 하는 메뉴 항목 및 칩 설계 검증의 동작 결과를 GUI 형태로 디스플레이한다.

<59> 또한 칩 설계 검증 프로그램(211)이 소프트웨어 IP(214)의 출력데이터를 인터페이스 수단(220)으로 전송하고, 인터페이스 수단(220)을 통해 입력되는 타겟(200)의 출력 데이터의 유효성 여부를 판단하고, 유효한 타겟(200)의 출력데이터만을 소프트웨어 IP(214)에 인가하도록 하도록 제어한다.

<60> 칩 설계 검증용 메모리(212)는 IP 입력용 레지스터(IPREG1), IP 출력용 레지스터(IPREG2), IP 입력용 메모리(IPMEM1), 및 IP 출력용 메모리(IPMEM2)를 구비하고, 칩 설계 검증 동작을 위해 필요한 데이터 및 칩 설계 검증 동작 시에 발생되는 각종 입출력 데이터를 저장 및 갱신한다. 또한 멀티 클럭 설정 값을 저장한다.

<61> 멀티 클럭 설정 값은 각 타겟 클럭의 하이레벨 점유시간 및 한 주기 점유 시간 값으로, 이러한 하이레벨 점유시간 및 한 주기 점유 시간 값은 인터페이스 수단(220)의 시스템 클럭 카운트 값으로 환산된 값이다.

<62> 테스트 벤치(213)는 칩 설계 검증에 필요한 테스트 케이스들과 클럭 패턴을 가지는 멀티 클럭을 발생한다.

<63> 소프트웨어 IP(214)는 타겟(200)과의 데이터 통신에 의해 동작을 수행한다.

<64> 인터페이스 수단(220)은 컴퓨터 시스템(2)의 확장 슬롯(170)에 설치되어 컴퓨터 시스템(2)의 외부에 존재하는 타겟(200)과의 연결을 도모하고, 타겟(200)의 출력데이터를 칩 설계 검증 프로그램(211)으로 전송하고, 칩 설계 검증 프로그램(211)을 통해 입력된 소프트웨어 IP(214)의 출력 데이터는 유효성 여부를 판단한 뒤, 유효한 소프트웨어 IP(214)의 출력데이터만을 타겟(200)으로 인가한다.

<65> 도 4는 본 발명의 실시예에 따른 인터페이스 수단의 구성을 도시한 것으로, 도면에 도시된 바와 같이 인터페이스 수단(220)은 메모리(400), 버스 종단 회로(410), 버스 스위치(420), 제어부(430), 클럭 신호 증폭부(440), 콘넥터(C1, C2)를 구비한다.

<66> 이상의 인터페이스 수단(220)은 인터페이스 보드의 구성을 나타내는 것으로, 인터페이스 보드는 도 1의 컴퓨터 시스템(2)의 확장 슬롯(170)에 장착된다.

<67> 도 4에 나타낸 각 블록의 기능을 설명하면 다음과 같다.

<68> 메모리(400)는 타겟 입력용 레지스터(M1), 타겟 출력용 레지스터(M2), 타겟 입력용 메모리(M3), 및 타겟 출력용 메모리(M4)를 구비하고, 칩 설계 검증 동작을 위해 필요한 데이터 및 칩 설계 검증 동작 시에 발생되는 각종 입출력 데이터를 이들 레지스터들 및 메모리들을 통해 저장 및 갱신한다.

<69> 타겟 입력용 레지스터(M1) 및 타겟 출력용 레지스터(M2)의 영역의 행의 크기는 제 1 콘넥터(C1)를 통하여 타겟(200)에 연결되는 입출력 신호의 수에 대응되어 결정되며, 타겟 입력용 메모리(M3) 및 타겟 출력용 메모리(M4)의 크기는 PCI 버스(450)를 통하여 전송되는 소프트웨어 IP(214)와 타겟(200)간의 입출력 데이터 용량에 대응되어 결정된다.

<70> 버스 종단 회로(410)는 타겟(200)과 인터페이스 수단(220)의 신호 연결 상태를 결정하고, 타겟(200)과 인터페이스 수단(220)간에 송수신되는 신호의 레벨을 인터페이스 해 준다.

<71> 버스 스위치(420)는 제어부(430)와 타겟(200)간의 신호를 상호 연결하는 기능을 수행한다. 인터페이스 수단(220)에 입력되는 데이터에는 대부분의 경우, 타겟(200)으로 전송되어질 데이터 즉, 타겟의 입력 데이터와 타겟(200)으로부터 전송되는 데이터 즉, 타겟의 출력 데이터가 혼재되게 된다. 이에 버스 스위치(420)는 타

겟의 입력 데이터와 타겟의 출력 데이터를 구분하여 타겟의 입력 데이터는 타겟의 입력용 메모리에 대응시키고, 타겟의 출력 데이터는 타겟의 출력용 메모리에 대응될 수 있도록 스위칭 하여 준다.

<72> 제어부(430)는 시스템 메모리(130)와 메모리(400)사이, 메모리(400)와 타겟(200)사이, 타겟(200)과 메모리(400)사이, 및 메모리(400)와 시스템 메모리(130)사이의 데이터 전송을 제어하고, 타겟(200)을 동작시키기 위한 멀티 클럭을 발생하여 타겟(200)에 제공한다.

<73> 클럭 신호 증폭부(440)는 제어부(430)를 통해 발생된 멀티 클럭을 타겟(200)의 동작 신호 레벨로 증폭하여 타겟(200)으로 제공한다.

<74> 제 1 콘넥터(C1)는 인터페이스 수단(220)과 타겟(200)의 물리적인 연결을 위한 콘넥터로서, 이를 위해 외부의 타겟 연결 케이블을 통하여 타겟(200)과 연결하고, GUI를 통하여 신호에 상응하는 핀 대 신호(pin to signal) 지정을 수행한 뒤에 이를 통해 타겟(200)과 인터페이스 수단(220) 간의 데이터를 전송을 수행한다.

<75> 제 2 콘넥터(C2)는 타겟(200)과의 연결을 수행하고, 제어부(430)를 통해 발생된 멀티 클럭을 타겟(200)에 제공하여 준다.

<76> 도 5는 도 4에 나타낸 제어부의 실시예의 블록도로서, 제어부(430)는 리셋 처리부(550), 플러그 & 플레이 제어부(552), PCI 로컬 버스 인터페이스 제어부(554), 어드레스 발생부(556), 메모리 제어부(558), 범용 상태 레지스터 및 범용 상태 레지스터 제어부(560), 인터럽트 제어부(562), 트리거 조건 제어부(564), 클록 제어부(566), 리드-백/JTAG 제어부(568), 병/직렬 변환부(570), 데이터 압축/복

원 제어부(572), 타겟 인터페이스 제어부(574), 버스 스위치 및 종단 회로 제어부(576), 글리치(glitch) 검출부(578), 및 변화 발생 감지부(580)로 구성되어 있다.

<77> 도 5에 나타낸 블록들 각각의 기능을 설명하면 다음과 같다.

<78> 리셋 처리부(550)는 인터페이스 수단(220)의 제어부(430)의 내부 레지스터들에 대한 초기화 동작을 수행할 목적으로 사용되는 신호이다. 리셋 신호는 외부의 리셋 스위치에 의해서 강제적으로 리셋 되거나, 칩 설계 검증 프로그램(211)에 통해 리셋 되거나, PCI 버스 시스템에 의해서 리셋 되거나, 정상적인 동작 완료에 의해서 마지막 프레임 전송 완료시에 발생된다.

<79> 플러그 & 플레이 제어부(552)는 플러그 & 플레이 동작을 위한 각종 PCI 구성 레지스터들(미도시)에 대한 제어를 수행한다.

<80> PCI 로컬 버스 인터페이스 제어부(554)는 각종 PCI관련 신호와 직접적으로 연결되어 있는 부분으로, PCI 버스를 통한 시스템 메모리(130)와 인터페이스 수단(220)의 메모리(400)사이의 인터페이스 제어를 수행한다.

<81> 어드레스 발생부(556)는 컴퓨터 시스템(2)의 시스템 메모리(130)와 인터페이스 수단(220)의 메모리(400)사이의 데이터 전송시 또는 인터페이스 수단(220)의 메모리(400)와 타겟(200)사이의 데이터 전송시에 인터페이스 수단(220)의 메모리(400)에 대한 어드레스를 발생한다.

<82> 메모리 제어부(558)는 PCI 로컬 버스 인터페이스 제어부(554) 및 타겟 인터페이스 제어부(574)의 제어신호에 따라 인터페이스 수단(220)의 메모리(400)를 직

접적으로 억세스하여 데이터 전송 동작을 수행한다.

<83> 범용 레지스터 및 범용 레지스터 제어부(560)는 데이터 전송 동작에 따른 인터페이스 수단(220)의 각종 상태를 점검하고, 그 결과를 내부의 범용 상태 레지스터에 저장한다. 또한 멀티 클럭 설정 값을 저장한다.

<84> 인터럽트 제어부(562)는 컴퓨터 시스템(2)의 시스템 메모리(130)와 인터페이스 수단(220)의 메모리(400)사이의 데이터 전송을 요청하기 위한 인터럽트 발생을 제어한다.

<85> 트리거 조건 제어부(564)는 고속 로직 분석기의 동작 구현을 위하여 높은 샘플링 주파수에 의해서 동작하며, 칩 설계 검증 수단(210)이 동작됨에 의해서 화면상에 디스플레이 되는 각종 GUI에 의해서 장치 사용자가 데이터 압축/복원 제어부(572)를 통하여 입/출력되는 임의의 I/0들에 대하여 설정한 트리거 조건을 전송받은 상태에서 타겟(200)에 대한 입력신호와 출력신호를 실시간으로 모니터링하면서 트리거 조건에 일치하는 위치를 검출한다.

<86> 또한, 트리거 조건 제어부(564)는 장치 사용자가 GUI를 통하여 설정한 입력 또는 출력 프레임 스텝 수 또는 클럭 스텝 수를 트리거 조건으로 설정할 수도 있다. 입력 또는 출력 프레임 스텝 수가 트리거 조건 제어부(564)에 트리거 조건으로 설정된 경우에는 데이터 압축/복원 제어부(572)의 입/출력 처리 과정에서 카운팅되는 입력 또는 출력 프레임 수가 트리거 조건으로 설정된 입력 또는 출력 프레임 스텝 수와 일치되는 지점에서 트리거 동작을 수행하게 된다. 동일한 방법으로, 클럭 스텝 수가 트리거 조건으로 설정된 경우에는 클럭 제어부(568)로부터 발생되

는 클럭 신호들 가운데 트리거 조건으로 설정된 클럭 신호의 카운팅 결과가 트리거 조건으로 설정된 클럭 스텝 수와 일치되는 지점에서 트리거 동작을 수행한다. 트리거 조건은 상술한 바와 같은 방법으로 개별적으로 적용하거나 우선순위를 두어서 혼용하여 적용할 수도 있다. 상술한 방법에 의해서 트리거 위치가 검출되면, 트리거 조건 제어부(564)는 장치 사용자에 의해서 설정된 GUI상의 디스플레이 기준 위치를 고려하여 어드레스 발생부(556)와 메모리 제어부(558)로부터 입력되는 트리거 지점에서의 어드레스와 메모리 영역 정보로부터 화면 디스플레이를 위한 시작 어드레스와 시작 어드레스의 메모리 영역, 종료 어드레스와 종료 어드레스의 메모리 영역에 관한 정보를 범용상태 레지스터 제어부(560)로 저장하고, 인터럽트 제어부(562)에 디스플레이 해당 메모리 구간의 데이터 전송을 알리는 인터럽트 요청 신호를 전달한다. 또한, 장치 사용자에 의해서 지정된 GUI상의 표시 위치를 고려하여 어드레스 발생부(556)와 메모리 제어부(558)에 대한 기준 신호를 발생한다.

<87> 클럭 제어부(568)는 인터페이스 수단(220)의 시스템 클럭의 속도를 체배하는 적어도 하나 이상의 위상 동기 루프(PLL : Phase Locked Loop)를 구비하고, 장치 사용자에 의해 선택에 따라 시스템 클럭의 속도를 선택한다.

<88> 만약, 인터페이스 수단이 복수개의 PLL을 구비하는 경우, 장치 사용자는 각 단의 PLL의 출력 주파수를 선택할 수 있으며, 인터페이스 수단(220)은 선택된 단의 PLL의 출력 주파수를 시스템 클럭으로 설정한다.

<89> 또한 범용 레지스터(560)에 저장된 멀티 클럭 설정 값을 획득하고, 장치 사용자에 의해 선택된 시스템 클럭 카운트 값을 기준하여, 타겟(220)으로 멀티 클럭

을 발생한다.

<90> 리드-백/JTAG 제어부(566)는 타겟(200)을 구성하는 FPGA, MCU, 기타 리드-백/JTAG을 지원하는 소자들의 임의의 내부 노드 또는 레지스터들에 대한 값을 읽어낸다. JTAG에 관한 사항은 IEEE 1149.1에 따른다. GUI의 리드-백/JTAG 관련 메뉴 항목을 이용하여 장치 사용자가 설정한 리드-백/JTAG 수행 대상 소자들의 내부 노드 또는 레지스터들의 위치 정보, 연속적인 동작의 수행을 위한 클럭 스텝의 수 등과 같은 정보를 전달 받은 뒤에, 이를 기초로 하여 타겟(200)을 구성하는 해당 소자에 대한 리드-백/JTAG 입력 신호와 출력 신호에 대한 발생과 저장을 제어한다.

<91> 병/직렬 변환부(570)는 직렬 데이터를 병렬로 변환하거나, 병렬 데이터를 직렬 데이터로 변환한다.

<92> 데이터 압축/복원 제어부(572)는 인터페이스 수단(220)의 메모리(400)에 압축된 형태로 저장되어 있는 데이터를 타겟(200)으로 인가할 때 복원하는 동작을 수행하고, 타겟(200)으로부터 인가되는 데이터를 저장할 때 압축하는 동작을 수행한다. GUI상에서 사용자의 지정에 의하거나 데이터 전송레이트(Rate)를 산출하는 방법에 의하여 변화 발생 감지부(580)를 통하여 검출된 타겟(200)의 출력단의 변화된 값은 전체적으로 압축이 행하여 지기 보다는 변화된 부분의 위치값과 변화된 값만을 추출하여 압축 동작을 수행한 뒤에 타겟 인터페이스 제어부(574)로 넘겨질 수 있다.

<93> 타겟 인터페이스 제어부(574)는 칩 검증 프로그램(211)가 수행됨에 의해 소프트웨어 IP(214)의 출력 데이터를 전송받으면, 출력 데이터의 유효 및 무효 여부

를 판단한 후, 유효한 데이터만을 타겟(200)으로 인가한다.

<94> 또한 타겟 인터페이스 제어부(574)는 변화 발생 감지부(580)를 통해 타겟(200) 출력단의 값의 변화 발생을 통보받으면, 변화된 타겟의 출력값만 추출하고, 이를 출력 데이터로서 컴퓨터 시스템(2)으로 전송하여 준다.

<95> 버스 스위치 및 종단 회로 제어부(576)는 칩 설계 검증 프로그램(211)의 핀 대 신호 지정 결과를 전달받아 타겟(200)에 대한 연결동작에 대한 제어를 수행하며, 각종 GUI 상에서 연결 해제 상태(tri-state)를 지정한 경우에는 버스 종단 회로(410)가 전기적으로 분리된 상태를 유지하도록 하여 콘넥터(C1)와 버스 종단 회로(410)사이에 데이터가 전송되지 못하도록 한다.

<96> 글리치 검출부(578)는 타겟(200)으로부터 유입되는 데이터에 포함된 글리치 성분을 검출하고, 글리치 성분의 위치를 모니터(1) 상에 표시하기 위하여 글리치가 검출된 지점에서의 클럭 수 또는 인터페이스 수단(220)의 메모리(400)의 어드레스를 별도의 레지스터(미도시)나 메모리(400)에 저장한다.

<97> 변화 발생 감지부(580)는 시스템 클럭에 따라 지속적으로 타겟(200)의 출력단의 값의 변화를 감지하고, 타겟(200)의 출력단의 값의 변화를 데이터 압축/목원 제어부(572)와 타겟 인터페이스 제어부(574)에 통보한다.

<98> 도 6은 본 발명의 실시예에 따른 칩 설계 검증 방법을 수행함에 따른 컴퓨터 시스템의 동작으로 설명하기 위한 흐름도이다.

<99> 장치 사용자가 칩 설계 검증 프로그램(211)을 실행시킨다. 이에 CPU(100)는

하드 디스크(140)에 저장된 칩 설계 검증 프로그램(211)을 시스템 메모리(130)상에 상주시키고, 그래픽 처리 장치(110)를 통하여 칩 설계 검증 프로그램(211)의 GUI를 모니터(1)상에 디스플레이 한다(제 110 단계).

<100> 장치 사용자는 디스플레이된 칩 설계 검증 프로그램(211)의 GUI상의 메뉴항목을 이용하여 각종 변수들에 대한 초기값 설정, 장치 사용자 지정 변수 및 관련 변수의 내용을 지정한다(제 120 단계).

<101> 제 120 단계에서 장치 사용자는 컴퓨터 시스템(2)의 시스템 메모리(130)의 메모리 영역 및 레지스터의 영역의 크기, 및 인터페이스 수단(220)의 메모리(400)의 메모리 영역 및 레지스터의 영역의 크기를 결정한다.

<102> 또한 인터페이스 수단(220)의 식별 정보, PCI 버스를 점유하고 있는지에 대한 정보, 시스템 메모리(130)와 인터페이스 수단(220) 사이의 신호 전송 방향, 억세스 되는 메모리, 전송 요구를 위한 인터럽트, 전송 완료를 위한 인터럽트, 및 리셋 정보 등을 설정한다.

<103> GUI상의 메뉴 항목을 이용하여 칩 설계 검증을 수행하고자 하는 소트웨어 IP(214)를 선택하여, 소프트웨어 IP(214)를 시스템 메모리(130)상에 상주시킨다(제 130 단계).

<104> GUI상의 신호- 핀 매핑 창을 띄우고, 선택된 소프트웨어 IP(214)의 입출력 신호를 타겟(200)상의 해당 입출력 핀과 매핑 관계를 형성한다(제 140 단계).

<105> GUI상의 메뉴항목을 이용하여 소프트웨어 IP(214) 및 타겟(200)의 칩 설계

검증을 위한 멀티 클럭 및 시스템 클럭을 설정한다(제 150 단계).

<106> 제 150 단계에 대한 상세한 설명은 이하에서 상세히 설명하기로 한다.

<107> 선택된 소프트웨어 IP(214)를 이용하는 타겟(200)의 전원을 온 시키고(제 160 단계), GUI상의 메뉴항목을 이용하여 인터페이스 수단(220)의 버스 종단 회로(410)의 상태를 연결 상태로 설정한다(제 170 단계).

<108> 그리고 GUI상의 메뉴 항목 또는 별도의 외부 입력 장치(미도시)를 이용하여 타겟(200)에 동작 시작을 명령하여 준다(제 180 단계).

<109> 제 180 단계에서, GUI상의 메뉴 항목을 이용하여 타겟(200)의 동작 시작을 명령하는 경우, 장치 사용자는 타겟(200)의 동작 모드를 설정하여 줄 수 있다.

<110> 타겟(200)의 동작 모드에는 장치 사용자가 설정한 동작 시간(run-time)동안만 동작을 수행하도록 하는 제한 동작 모드와, 장치 사용자가 GUI상의 메뉴 항목 또는 별도의 외부 입력 장치(미도시)를 이용하여 타겟(200)의 정지 동작이 요청할 때까지 계속적으로 동작을 수행하는 무한 동작 모드가 있다.

<111> GUI상의 메뉴 항목 또는 별도의 외부 입력 장치(미도시)를 이용하여 소프트웨어 IP(214)를 포함한 타겟(200)이 초기화 동작을 요청받으면, 칩 설계 검증 프로그램(211)은 타겟의 초기화 신호를 인터페이스 수단(220)으로 전송(A)한다(제 190 단계).

<112> 도 7a는 도 6의 멀티 클럭 및 시스템 클럭의 설정 과정을 상세히 설명하기 위한 도면이고, 도 7b는 도 7a의 멀티 클럭 및 시스템 클럭의 다이아 그램을 나타

내는 도면이다.

<113> 멀티 클럭은 타겟(200) 및 소프트웨어 IP(214)의 동작을 위해 인가되는 적어도 하나 이상의 클럭으로 구성된다.

<114> 타겟(200) 및 소프트웨어 IP(214)는 복수개의 설계 블럭들(ex, 오디오, 비디오, 소프트웨어 IP 등)로 구성되며, 이들 설계 블럭들 각각은 서로 상이한 주기를 가지는 상이한 클럭을 인가받아 동작을 수행하게 된다.

<115> 시스템 클럭은 컴퓨터 시스템(2)에 내부적으로 위치하는 소프트웨어 IP(214)와 인터페이스 수단(220)과 연결되는 타겟(200)의 상호 동기용 클럭으로, 시스템 클럭 카운트 값을 기준으로 하여 소프트웨어 IP(214) 및 타겟(200)의 동작 시간(run-time)을 파악한다.

<116> 소프트웨어 IP(214)에 인가되는 시스템 클럭의 속도는 컴퓨터 시스템(2)의 내부 동작 속도에 의해 결정되며, 타겟(200)에 인가되는 시스템 클럭의 속도는 장치의 장치 사용자에 의해 선택된 인터페이스 수단(220)의 시스템 클럭 속도에 의해 결정된다.

<117> 장치 사용자는 먼저 GUI상의 메뉴 항목 또는 테스트 벤치를 이용하여, 칩 설계 검증에 필요한 멀티 클럭을 설정한다(제 151 단계).

<118> 제 151 단계에서 GUI상의 메뉴 항목을 이용하여 멀티 클럭을 정의하는 경우, 장치 사용자는 타겟(200) 및 소프트웨어 IP(214)의 동작을 필요한 클럭들을 선택하고, 각 선택된 클럭의 동작 속도를 정의한다. 이에 칩 설계 검증 프로그램

(211)은 칩 설계 검증을 위한 멀티 클럭을 구성하는 클럭들의 종류 및 각 클럭의 동작 속도를 획득한다.

<119> 예를 들어, 타겟(200) 및 소프트웨어 IP(214)가 66.1MHz, 47.2MHz, 및 850KHz인 타겟 클럭들이 필요하면, 장치 사용자는 GUI상의 메뉴 항목을 띄워, 제 1, 제 2, 및 제 3 타겟 클럭들을 선택하고, 제 1 타겟 클럭의 값에 850KHz를 입력하고, 제 2 타겟 클럭의 값에 47.2MHz을 입력하고, 제 3 타겟 클럭의 값에 66.1MHz을 입력하여 멀티 클럭을 설정한다.

<120> 여기서 개별 클럭의 값과 단위는 절대 클럭 값을 의미하는 것이 아니고 각 개별 클럭간의 비율을 상대 클럭 값을 의미한다.

<121> 또한 제 151 단계에서 테스트 벤치를 이용하여 멀티 클럭을 정의하는 경우, 칩 설계 검증 프로그램(211)은 테스트 벤치(213)의 클럭 정의 부분을 획득하여, 칩 설계 검증을 위한 멀티 클럭을 구성하는 클럭들의 종류 및 각 클럭의 동작 속도를 획득한다.

<122> 이때 일반적인 테스트 벤치는 멀티 클럭을 발생하는 클록 발생 블록과 신호 발생 블록으로부터 멀티 클럭을 제공받아 테스트 케이스를 발생하는 테스트 케이스 발생 블록을 포함한다.

<123> 이에 칩 설계 검증 프로그램(211)은 클록 발생 블록으로부터 클럭 정의 부분을 획득한 후, 클록 발생 블록과 테스트 케이스 발생 블록과의 연결을 해제하고, 칩 설계 검증 프로그램(211)과 테스트 케이스 발생 블록을 연결한다. 그러면 테스트 케이스 발생 블록은 칩 설계 검증 프로그램(211)의 멀티 클럭에 응답하여 해당

동작을 수행하여 준다.

<124> 제 153 단계에서 칩 설계 검증 프로그램(211)은 사용자에 의해 설정되거나 테스트 벤치로부터 획득된 타겟 클럭들 중 가장 느린 주기를 가지는 타겟 클럭(제 1 타겟 클럭)을 기준으로 설정하고, 이에 대한 나머지 타겟 클럭들(제 2 타겟 클럭 ~ 제 n 타겟 클럭)의 주기의 비율을 계산한다(제 153 단계).

<125> 도 7b의 (a)와 같은 칩 설계 검증 프로그램(211)의 클럭 표시창을 모니터(1)상에 띄운 후, 먼저 기준으로 설정된 타겟 클럭(제 1 타겟 클럭)의 한 주기 클럭을 클럭 표시창에 디스플레이 하고, 이를 기준으로 하여 나머지 타겟 클럭들(제 2 타겟 클럭 ~ 제 n 타겟 클럭)을 각각 계산된 비율에 해당하는 정도로 디스플레이 한다(제 155 단계).

<126> 장치 사용자는 GUI 상의 메뉴항목을 이용하여 디스플레이된 타겟 클럭들에 대하여 클럭 상호간의 위치 지정(P2, Pn)과 클럭 듀티(duty)(D1, D2, Dn)와 같은 타겟 클럭의 속성을 지정한다(제 157 단계).

<127> 각 타겟 클럭의 하이레벨 점유 기간 및 한주기 점유기간을 시스템 클럭카운트 값으로 환산하여 멀티 클럭 설정 값을 획득하고, 이를 인터페이스 수단(220)에 제한다(제 159 단계).

<128> 제 159 단계에서 장치 사용자는 인터페이스 수단(220)의 시스템 클럭의 속도를 선택하고, 선택된 속도에 대응되는 속도를 가지는 시스템 클럭을 도 7b의 (b)에 도시된 바와 같이 칩 설계 검증 프로그램(211)의 클럭 표시창에 디스플레이한다.

<129> 칩 설계 검증 프로그램(211)은 멀티 클럭을 구성하는 각 타겟 클럭의 하이레벨 점유 기간 및 한주기 점유기간을 시스템 클럭카운트 값으로 환산하여 멀티 클럭 설정 값을 획득하고, 이를 컴퓨터 시스템(2)의 시스템 메모리(130)의 칩 설계 검증용 메모리(212)와 인터페이스 수단(220)의 제어부(430)의 범용 레지스터(560)에 저장한다.

<130> 칩 설계 검증 프로그램(211)은 실제적인 칩 설계 검증 동작시, 칩 설계 검증용 메모리(212)에 저장된 멀티 클럭 설정 값과 컴퓨터 시스템(2)의 시스템 클럭을 기반으로 하여 소프트웨어 IP(214)의 각 기능 블록들을 동작시키기 위한 멀티 클럭을 발생하여 소프트웨어 IP(214)에 인가한다.

<131> 또한 인터페이스 수단(220)의 제어부(430)의 클럭 제어부(568)는 범용 레지스터(560)에 저장된 멀티 클럭 설정 값과 장치 사용자에 의해 선택된 속도에 대응되는 인터페이스 수단(220)의 시스템 클럭을 기반으로 하여 타겟(200)의 각 기능 블록들을 동작시키기 위한 멀티 클럭을 발생한다.

<132> 이상에서 살펴본 바와 같이 본 발명에서는 인터페이스 수단(220)이 독립적으로 타겟(200)을 동작시키기 위한 멀티 클럭을 발생하여 타겟(200)에 인가한다.

<133> 이에 본 발명에서는 종래의 기술에서 발생하는 인터페이스 수단(200)의 동작 지연에 의한 데이터의 전송 지연 즉, 컴퓨터 시스템(2)내에서 존재하는 테스트 벤치(213)로부터 타겟(200) 인가용 멀티 클럭을 제공받음으로서 발생하는 전송 지연을 획기적으로 감소시켜 줄 수 있게 된다.

<134> 도 8과 도 9a 내지 9b는 본 발명의 실시예에 따른 칩 설계 검증 방법을 설명하기 위한 흐름도로, 도 8은 본 발명의 실시예에 따른 칩 설계 검증 방법의 초기화 동작을 설명하기 위한 흐름도이고, 도 9a는 본 발명의 실시예에 따른 칩 설계 검증 방법의 칩 설계 검증 프로그램의 소프트웨어 IP에 대한(이하에서는 IP측이라 지칭한다) 메인 동작을 설명하기 위한 흐름도이고, 도 9b는 본 발명의 실시예에 따른 칩 설계 검증 방법의 칩 설계 검증 프로그램의 인터페이스 수단의 타겟에 대한(이하에서는 타겟측이라 지칭한다) 메인 동작을 설명하기 위한 흐름도이다.

<135> 본 발명의 실시예에 따른 칩 설계 검증 방법은 도면에 도시된 바와 같이, IP측의 동작과 타겟측의 동작이 상호 동작되면서 초기화 동작 및 메인 동작이 수행되게 된다.

<136> 먼저, 도 8을 참조하여 본 발명의 실시예에 따른 칩 설계 검증 방법의 초기화 동작을 살펴보도록 하자.

<137> 칩 설계 검증 프로그램(211)의 초기화 동작이 수행됨에 의해 인터페이스 수단(220)이 타겟의 초기화 신호를 수신하게 되면(제 210 단계), 인터페이스 수단(220)은 수신된 타겟의 초기화 신호를 타겟(200)에 전송한다.

<138> 이에 타겟(200)은 수신한 타겟의 초기화 신호에 응답하여 초기화를 수행하여 초기화된 타겟(200)의 출력값을 발생한다(제 220 단계).

<139> 인터페이스 수단(220)은 타겟(200)이 초기화된 타겟(200)의 출력값을 발생함이 감지되면, 인터페이스 수단(220)은 소프트웨어 IP의 초기화 신호를 생성하고,

"타겟 클럭 카운트 값"을 0으로 설정한다. 그리고 생성된 소프트웨어 IP의 초기화 신호와, 0으로 설정된 "타겟 클럭 카운트 값" 및 발생된 초기화된 타겟(200)의 출력값으로 구성된 타겟의 출력 데이터를 IP측으로 전송한다(제 230 단계).

<140> 제 230 단계에서 인터페이스 수단(220)은 타겟의 출력 데이터를 IP측으로 전송하기 위해, 먼저 인터페이스 수단(220)의 제어부(430)의 데이터 압축/ 복원 제어부(572)를 통해 "타겟의 출력값"을 압축하고, 압축된 "타겟의 출력값"과 "타겟 클럭 카운트 값"으로 구성된 출력 데이터를 생성한다.

<141> 그리고 생성된 출력 데이터를 인터페이스 수단(220)의 메모리(400)의 타겟 출력용에 저장한 후, 인터럽트 제어부(562)를 통해 전송 요청 인터럽트 신호를 발생하여 IP측으로 전송한다.

<142> 이에 칩 설계 검증 프로그램(211)은 인터페이스 수단(220)의 제어부(430)의 PCI 버스 인터페이스 제어부(554)를 제어하여 PCI 버스(450)를 통해 컴퓨터 시스템(2)의 칩 설계 검증용 메모리(212)의 IP 입력용 메모리(IPMEM1) 영역에 타겟의 출력 데이터를 저장한다.

<143> 또한 인터페이스 수단(220)은 IP측으로부터 초기화된 소프트웨어 IP(214)의 출력 데이터를 수신하는 지를 모니터링 하여 대기하고, 마침내 IP측으로부터 소프트웨어 IP(214)의 출력 데이터가 전송되면(제 240 단계), 이에 대한 인식 신호(Acknowledge signal ; 이하 "ACKN 신호"라 지칭한다)를 발생하여 IP측으로 전송하고(제 250 단계), 타겟(200)의 초기화 동작이 정상적으로 수행하였음을 확인하고 타겟측의 메인 동작으로 진입(C)한다.

<144> 칩 설계 검증 프로그램(211)은 제 230 단계에 의해 전송된 인터페이스 수단(220)의 초기화된 타겟(200)의 "타겟 클럭 카운트 값"과 "타겟의 출력값"으로 구성되는 출력 데이터와 소프트웨어 IP의 초기화 신호를 수신하면(제 260 단계), 수신된 타겟(200)의 출력값과 소프트웨어 IP의 초기화 신호를 소프트웨어 IP(214)에 입력한다

<145> 제 260 단계에서 칩 설계 검증 프로그램(211)은 수신된 "타겟의 출력값"을 소프트웨어 IP(214)에 입력하기 위해, 먼저 IP 입력용 메모리(IPMEM1)에 압축되어 저장된 "타겟의 출력값"을 복원하고, 복원된 "타겟의 출력값"을 IP 입력용 레지스터(IPREG1)에 저장한다. 이어서, IP 입력용 레지스터(IPREG1)에 저장된 "타겟의 출력값"은 소프트웨어 IP(214)의 입력단으로 인가된다.

<146> 소프트웨어 IP(214)는 수신된 초기화된 타겟(200)의 출력값과 소프트웨어 IP의 초기화 신호에 응답하여 초기화를 수행하여 초기화된 소프트웨어 IP(214)의 출력값을 발생한다(제 270 단계).

<147> 칩 설계 검증 프로그램(211)은 "IP 클럭 카운트 값"을 0으로 설정하고, 0으로 설정된 "IP 클럭 카운트 값' 및 발생된 초기화된 "IP의 출력 값"을 타겟측으로 전송한다(제 280 단계).

<148> "IP의 출력 값"은 전송한 초기화된 "IP의 출력 값"에 대한 "ACKN 신호"가 타겟측으로부터 전송되는 지를 모니터링 하여 대기하고, 마침내 타겟측으로부터 "ACKN 신호"가 전송되면(제 290 단계), 초기화 동작을 정상적으로 수행하였음을 확인하고 IP측의 메인 동작으로 진입(D)한다.

<149> 이상과 같은 단계를 통해 소프트웨어 IP(214) 및 타겟(200)의 초기화 동작이 성공적으로 완료되면, 이 시점부터 타겟측과 IP측 각각은 메인 동작을 수행하는 단계로 접어든다.

<150> 먼저, 도 9a를 참조하여 타겟측의 메인동작을 살펴보면 다음과 같다.

<151> 상술한 바와 같이 타겟(200)의 초기화 동작이 정상적으로 완료되어, 메인 동작으로 진입(C)을 수행하게 되면, 인터페이스 수단(220)은 타겟(200)에 인터페이스 수단(220)의 시스템 클럭을 한 클럭 인가하고(제 370 단계), "타겟 클럭 카운트 값"을 1 만큼 증가시키면서(제 380 단계), 변화 발생 감지부(580)을 통하여 "타겟의 출력값"이 변화되는 지를 모니터링 한다(제 390 단계).

<152> 제 390 단계의 모니터링 결과, "타겟의 출력값"이 변화되지 않았으면, "IP 클럭 카운트 값과 비교 플래그"가 설정된 상태 인지를 확인하는 단계(제 400 단계)를 거쳐 다시 제 370 단계로 진행하여 준다.

<153> 인터페이스 수단(220)은 제 370 단계에서 제 400단계 사이를 반복 수행하면서 "타겟의 출력값"에 변화되었는지를 모니터링하면서 IP측으로부터의 진입(G)을 대기한다.

<154> 이에 "타겟의 출력값"이 변화되기 이전에 IP측으로부터의 진입(G)이 확인되면, 즉, IP측으로부터 전송된 소프트웨어 IP(214)의 출력데이터가 인터페이스 수단(220)에 수신되면(제 310 단계), 인터페이스 수단(220)의 제어부(430)는 "IP 클럭 카운트 값"을 획득하고, IP측으로부터 전송된 "IP 클럭 카운트 값"이 현재까지 누

적된 "타겟 클럭 카운트 값" 보다 작거나 같은지를 확인한다(제 320 단계).

<155> 제 320 단계의 확인 결과, "IP 클럭 카운트 값"이 "타겟 클럭 카운트 값" 보다 작거나 같으면, IP측으로 "ACKN 신호"를 전송하고(제 340 단계), 인터페이스 수단(220)은 "IP의 출력값"을 타겟(200)의 입력 데이터로 갱신한다(제 345 단계).

<156> 그리고 "타겟 클럭 카운트 값"을 초기화한다(제 350 단계).

<157> 또한 인터페이스 수단(220)의 제어부(430)는 갱신된 타겟(200)의 입력 데이터가 타겟(200)의 해당 입력에 인가될 수 있도록 데이터 압축/복원 제어부(572)를 통해 압축된 "소프트웨어 IP의 출력값"을 복원한 후, 타겟 입력용 레지스터(M1)에 저장한다(제 360 단계).

<158> 그리고 인터페이스 수단(220)은 제 370 단계에서 제 400 단계 사이를 반복 수행하면서 "타겟의 출력값"에 변화가 발생되었는지를 모니터링하면서, 새로운 IP 측으로부터 진입(G)을 대기한다.

<159> 반면에 제 320 단계의 확인 결과, "IP 클럭 카운트 값"이 "타겟 클럭 카운트 값"보다 크다면, "IP 클럭 카운트 값"과 "타겟 클럭 카운트 값"을 비교하기 위한 "IP 클럭 카운트 값과 비교 플래그"를 설정하고(제 330 단계), 현재까지 누적된 "타겟 클럭 카운트 값"이 "IP 클럭 카운트 값"보다 큰지를 확인하는 단계(제 410 단계)로 진행한다.

<160> 제 410 단계의 확인결과, "타겟 클럭 카운트 값"이 "IP 클럭 카운트 값"보다 작거나 같으면, 다시 제 370 단계에서 제 400 단계 사이를 반복 수행하면서 "타겟

의 출력값"에 변화가 발생되는지를 모니터링한다.

<161> 이에 "타겟 클럭 카운트 값"이 "IP 클럭 카운트 값"보다 작은 상태에서 "타겟의 출력값"에 변화발생이 감지되면, 인터페이스 수단(220)은 변화된 "타겟의 출력값"과 이때의 누적된 "타겟 클럭 카운트 값"으로 구성된 타겟(200)의 출력데이터를 IP측으로 전송하여(제 430 단계), IP측으로 진입(E)을 수행한다.

<162> 그리고 인터페이스 수단(220)은 IP측으로부터 전송한 데이터에 대한 "ACKN 신호"를 수신되는지를 모니터링 하면서 대기하고, IP측으로부터 "ACKN 신호"가 수신되면(제 440 단계), 곧바로 제 340 단계로 진입(F)을 수행한다.

<163> 반면에 제 410 단계의 확인결과, "타겟 클럭 카운트 값"이 "IP 클럭 카운트 값"보다 클 때까지 "타겟의 출력값"이 변화되지 않으면, 인터페이스 수단(200)은 "IP 클럭 카운트 값과 비교 플래그"를 해제하고(제 420 단계), 전송된 소프트웨어 IP(214)의 출력 데이터를 새로운 입력 데이터로 갱신하는 단계(제 340 단계)로 진행한다.

<164> 다음으로 도 9b를 참조하여 IP측의 메인 동작을 살펴보도록 하자.

<165> 상술한 바와 같이 소프트웨어 IP(214)의 초기화 동작이 정상적으로 완료되어, 메인 동작으로의 진입(D)을 수행하게 되면, 칩 설계 검증 프로그램(211)은 소프트웨어 IP(214)에 시스템 클럭을 한 클럭 인가하고(제 570 단계), "IP 클럭 카운트 값"을 1 만큼 증가시키면서(제 580 단계), "소프트웨어 IP의 출력값"에 변화가 발생되는 지를 모니터링 한다(제 590 단계).

<166> 제 590 단계의 모니터링 결과, "소프트웨어 IP의 출력값"에 변화가 발생되지 않았으면, "타겟 클럭 카운트 값과 비교 플래그"가 설정된 상태 인지를 확인하는 단계(제 600 단계)를 거쳐 다시 제 570 단계로 진행한다.

<167> 칩 설계 검증 프로그램(211)은 제 570 단계에서 제 600 단계 사이를 반복 수행하면서 "소프트웨어 IP의 출력값"에 변화가 발생되는지를 모니터링하고, 타겟측으로부터의 진입(E)을 대기한다.

<168> 이에 "소프트웨어 IP의 출력값"이 변화되기 이전에 타겟측으로부터 진입(E)이 확인되면 즉, 타겟측으로부터 전송된 타겟(200)의 출력데이터가 수신되면(제 510 단계), 칩 설계 검증 프로그램(211)은 타겟측으로부터 전송된 "타겟 클럭 카운트 값"이 현재까지 누적된 "IP 클럭 카운트 값" 보다 작거나 같은지를 확인한다(제 520 단계).

<169> 제 520 단계의 확인 결과, "타겟 클럭 카운트 값"이 "IP 클럭 카운트 값" 보다 작거나 같으면, 타겟측으로 "ACKN 신호"를 전송한다(제 540 단계).

<170> 그리고 칩 설계 검증 프로그램(211)은 "타겟의 출력값"을 소프트웨어 IP(214)의 새로운 입력 데이터로 갱신한다(제 542 단계).

<171> 그리고 칩 설계 검증 프로그램(211)은 IP 입력용 메모리(IPMEM1)에 압축 저장된 "타겟의 출력값"을 복원한 후, "타겟 클럭 카운트 값"에 따라 "타겟의 출력값" 파형관측기 창에 디스플레이 한다. 즉 파형 관측창에 타겟(200)에 대한 출력 파형을 갱신한다(제 545 단계).

<172> 칩 설계 검증 프로그램(211)은 "IP 클럭 카운트 값"을 초기화하고(제 550 단계), 갱신된 입력 데이터가 소프트웨어 IP(214)의 해당 입력에 인가될 수 있도록 복원된 "타겟의 출력값"을 IP 입력용 레지스터(IPREG1)에 저장한다(제 560 단계).

<173> 그리고 칩 설계 검증 프로그램(211)은 제 570 단계에서 제 600 단계 사이를 반복 수행하면서 "소프트웨어 IP의 출력값"에 변화가 발생되는 지를 모니터링하면서, 새로운 타겟측으로부터의 진입(E)을 대기한다.

<174> 반면에 제 520 단계의 확인 결과, "타겟 클럭 카운트 값"이 "IP 클럭 카운트 값"보다 크다면, "IP 클럭 카운트 값"과 "타겟 클럭 카운트 값"을 비교하기 위한 "타겟 클럭 카운트 값과 비교 플래그"를 설정하고(제 530 단계), 현재까지 누적된 "IP 클럭 카운트 값"이 "타겟 클럭 카운트 값"보다 큰지를 확인하는 단계(제 610 단계)로 진행한다.

<175> 제 610 단계의 확인결과, "IP 클럭 카운트 값"이 "타겟 클럭 카운트 값"보다 작거나 같으면, 다시 제 570 단계에서 제 600 단계 사이를 반복 수행하면서 "소프트웨어 IP의 출력값"에 변화가 발생되는 지를 모니터링한다.

<176> 이에 "IP 클럭 카운트 값"이 "타겟 클럭 카운트 값"보다 작은 상태에서 "소프트웨어 IP의 출력값"에 변화가 감지되면, 동작의 중지 여부를 확인하기 위해 칩 설계 검증 프로그램(211)은 누적된 실효적인 것으로 "타겟 클럭 카운트 값"에 의한 동작 수행 시간이 장치 사용자가 설정한 시간(run-time)값과 일치하는지를 확인한다(제 624 단계).

<177> 장치 사용자가 GUI상의 메뉴항목 또는 외부의 입력 장치(미도시)를 통해 동작 중지를 요청하였음을 감지한다(제 628 단계).

<178> 제 624 단계 및 제 628 단계의 확인 결과, 칩 설계 검증 동작이 계속 수행되어야 하면, 칩 설계 검증 프로그램(211)은 변화된 "소프트웨어 IP의 출력값"과 이때의 누적된 "IP 클럭 카운트 값"으로 구성된 소프트웨어 IP(214)의 출력데이터를 타겟측으로 전송하여 타겟측으로의 진입(G)을 수행한다(제 630 단계).

<179> 그리고 칩 설계 검증 프로그램(211)은 타겟측으로부터 전송한 데이터에 대한 "ACKN 신호"가 수신되는 지를 모니터링 하면서 대기하고, 타겟측으로부터 "ACKN 신호"가 전달되면(제 640 단계), 누적된 "IP 클럭 카운트 값"에 따라 변화된 "소프트웨어 IP의 출력값"을 파형 관측창에 디스플레이 한다. 즉, 파형 관측창에 소프트웨어 IP(214)의 출력 파형을 갱신하고(제 645 단계), 제 540 단계로 진입(H)을 수행한다.

<180> 반면에 제 624 단계 및 제 628 단계의 확인 결과, 칩 설계 검증 동작이 중지되어야 하면 칩 설계 검증 프로그램(211)은 소프트웨어 IP(214)에 시스템 클럭의 인가를 중지하고, IP 클럭의 카운팅을 중지하고, 인터페이스 수단(220)을 제어하여 인터페이스 수단(220)이 타겟(200)에 시스템 클럭을 인가하지 않도록 한다(제 650 단계).

<181> 또한 제 610 단계의 확인결과, "IP 클럭 카운트 값"이 "타겟 클럭 카운트 값"보다 클 때까지 "IP의 출력값"에 변화가 발생하지 않으면, 칩 설계 검증 프로그램(211)은 "타겟 클럭 카운트 값과 비교 플래그"를 해제하고(제 620 단계), 전송된

타겟(200)의 출력 데이터를 새로운 입력 데이터로 갱신하는 단계(제 540 단계)로 진행한다.

<182> 도 8 내지 도9a 및 도 9b에서 설명되지는 않았지만, 칩 설계 검증 프로그램(211)과 인터페이스 수단(220)은 독립적인 동작에 의해 생성된 시스템 클럭을 소프트웨어 IP(214)와 타겟(200)에 각각 인가함과 동시에 시스템 클럭 카운트 값을 기준으로 하여 멀티 클럭을 생성하여 소프트웨어 IP(214)와 타겟(200)에 각각 인가하여 주는 것은 당연하다.

<183> 또한 도9a 및 도 9b에서 IP측 또는 타겟측에서 상대측인 타겟측 또는 IP측으로 출력 데이터를 전송하는 경우, 데이터 용량을 감소시키기 위해, 변화된 데이터만을 획득하고 압축하여 전송하는 것이 가능하다.

<184> 상기에서는 본 발명의 바람직한 실시예를 참조하여 설명하였지만, 해당 기술분야의 숙련된 당업자는 하기의 특허 청구의 범위에 기재된 본 발명의 사상 및 영역으로부터 벗어나지 않는 범위 내에서 본 발명을 다양하게 수정 및 변경시킬 수 있음을 이해할 수 있을 것이다.

【발명의 효과】

<185> 따라서 본 발명의 칩 설계 검증 및 방법은 칩 설계 검증 프로그램 및 인터페이스 수단을 이용하여 칩 설계 단계에서 칩 설계의 오류 및 에러를 검증할 수 있도록 하는 칩의 설계의 정확성을 제공하고, 칩 설계의 소요 시간 및 비용을 감소시켜 준다.

<186> 또한 칩 설계 검증 및 방법은 소프트웨어 IP 또는 타겟의 출력 데이터가 변화되었을 경우에만, 즉 이벤트가 발생한 경우에만 상대측과 데이터 송수신을 수행하도록 하여 데이터 전송 속도 및 효율을 증대한다.

<187> 또한 칩 설계 검증 및 방법은 데이터 전송 시, 소프트웨어 IP 또는 타겟의 변화된 출력 데이터를 추출하여 압축하고, 이를 상대측에 전송하도록 하여 데이터 전송 속도 및 효율을 증대한다.

<188> 또한 칩 설계 검증 및 방법은 인터페이스 수단이 독립적으로 타겟을 동작시키기 위한 멀티 클럭을 발생하여 직접 인가할 수 있도록 하여 컴퓨터 시스템내로부터 멀티 클럭을 제공받음으로써 발생하게 되는 딜레이를 획기적으로 감소시켜 준다. 이에 보다 빠른 칩 설계 검증 속도를 지원하여 준다.

【특허청구범위】

【청구항 1】

적어도 하나 이상의 하드웨어 블록; 및

상기 하드웨어 블록과 데이터 통신을 수행하는 적어도 하나 이상의 소프트웨어 블록을 포함하고, 상기 하드웨어 블록과 상기 소프트웨어 블록간의 동작을 검증하는 컴퓨터를 구비하고,

상기 컴퓨터는

상기 하드웨어 블록의 출력데이터를 전송하고, 상기 소프트웨어 블록의 출력데이터의 유효성 여부를 판단하고, 유효한 소프트웨어 블록의 출력데이터만을 상기 하드웨어 블록으로 인가하는 인터페이스 수단;

상기 소프트웨어 블록, 상기 하드웨어 블록과 상기 소프트웨어 블록간의 동작을 검증하는 칩 설계 검증 프로그램을 저장하는 저장수단; 및

상기 칩 설계 검증 프로그램을 수행하여 상기 소프트웨어 블록의 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 인터페이스 수단을 통해 입력되는 상기 하드웨어 블록의 출력 데이터의 유효성 여부를 판단하고, 유효한 하드웨어 블록의 출력데이터만을 상기 소프트웨어 블록에 인가하도록 하는 제어부를 구비하는 것을 특징으로 하는 칩 설계 검증 장치.

【청구항 2】

제 1 항에 있어서, 상기 칩 설계 검증 프로그램은

그래픽 유저 인터페이스를 구비하고, 상기 칩 설계 검증 프로그램을 수행함에 의해 송수신되는 데이터를 상기 그래픽 유저 인터페이스를 통해 디스플레이하는 것을 특징으로 하는 칩 설계 검증 장치.

【청구항 3】

제 1 항에 있어서, 상기 칩 설계 검증 프로그램은

상기 소프트웨어 블록 및 상기 하드웨어 블록을 동작시키기 위한 멀티 클럭 설정값을 획득하고, 상기 인터페이스 수단에 저장하는 것을 특징으로 하는 칩 설계 검증 장치.

【청구항 4】

제 3 항에 있어서, 상기 인터페이스 수단은

상기 멀티 클럭 설정 값과, 상기 인터페이스 수단의 시스템 클럭에 따라 멀티 클럭을 생성하여 상기 하드웨어 블록에 인가하는 클럭 제어부를 구비하는 것을 특징으로 하는 칩 설계 검증 장치.

【청구항 5】

제 4 항에 있어서, 상기 하드웨어 블록의 출력데이터는

상기 인터페이스 수단으로부터 인가된 멀티 클럭에 응답하여 변화된 상기 하드웨어 블록의 출력값과, 상기 하드웨어 블록의 출력값이 변화될 때의 상기 인터페이스 수단의 시스템 클럭 카운트 값으로 구성되는 것을 특징으로 하는 칩 설계 검증 장치.

【청구항 6】

제 5 항에 있어서, 상기 유효한 하드웨어 블록의 출력데이터는

상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작은 경우에는 상기 하드웨어 블록의 출력값이고, 상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커서 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않는 경우에는 상기 하드웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 장치.

【청구항 7】

제 3 항에 있어서, 상기 칩 설계 검증 프로그램은

상기 멀티 클럭 설정 값과 상기 칩 설계 검증 프로그램의 시스템 클럭에 따라 멀티 클럭을 생성하여 상기 소프트웨어 블록에 인가하는 것을 특징으로 하는 칩 설계 검증 장치.

【청구항 8】

제 7 항에 있어서, 상기 소프트웨어 블록의 출력데이터는

상기 칩 설계 검증 프로그램으로부터 인가된 멀티 클럭에 응답하여 변화된 상기 소프트웨어 블록의 출력값과, 상기 소프트웨어 블록의 출력값이 변화될 때의 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값으로 구성되는 것을 특징으로

하는 칩 설계 검증 장치.

【청구항 9】

제 8 항에 있어서, 상기 유효한 소프트웨어 블록의 출력데이터는

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 작은 경우에는 상기 소프트웨어 블록의 출력값이고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 커서 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 하드웨어 블록의 출력값이 변화하지 않는 경우에는 상기 소프트웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 장치.

【청구항 10】

제 7 항에 있어서, 상기 소프트웨어 블록은

테스트 벤치를 구비하고, 상기 테스트 벤치는 상기 멀티 클럭 설정 값을 상기 칩 설계 검증 프로그램에 제공하고, 상기 테스트 벤치의 자체적인 멀티 클럭 대신에 상기 칩 설계 검증 프로그램의 멀티 클럭에 응답하여 동작하는 것을 특징으로 하는 칩 설계 검증 장치.

【청구항 11】

적어도 하나 이상의 하드웨어 블록, 및 적어도 하나 이상의 소프트웨어 블록, 칩 설계 검증 프로그램, 및 입출력 데이터를 저장하는 저장수단과 상기 소프

트웨어 블록과 상기 하드웨어 블록의 인터페이스를 수행하는 인터페이스 수단을 구비하는 컴퓨터를 구비하는 칩 설계 검증 장치의 칩 설계 검증 방법에 있어서,

상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이스 수단으로 전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 유효한 하드웨어 블록의 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계; 및

상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 상기 유효한 소프트웨어 블록의 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 12】

제 11 항에 있어서,

상기 칩 설계 검증 프로그램은 멀티 클럭 설정값을 획득하여, 상기 인터페이스 수단에 제공하고, 상기 멀티 클럭 설정값과 상기 칩 설계 검증 프로그램의 시스템 클럭에 따라 멀티 클럭을 발생하여 상기 소프트웨어 블록에 인가하는 단계; 및

상기 인터페이스 수단은 상기 멀티 클럭 설정값과 상기 인터페이스 수단의 시스템 클럭에 따라 멀티 클럭을 발생하여 상기 하드웨어 블록에 인가하는 단계를

더 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 13】

제 12 항에 있어서, 상기 하드웨어 블록의 출력 데이터는

상기 인터페이스 수단으로부터 인가된 멀티 클럭에 응답하여 변화된 상기 하드웨어 블록의 출력값과, 상기 하드웨어 블록의 출력값이 변화될 때의 상기 인터페이스 수단의 시스템 클럭 카운트 값으로 구성되는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 14】

제 13 항에 있어서, 상기 유효한 하드웨어 블록의 출력 데이터는

상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작은 경우에는 상기 하드웨어 블록의 출력값이고, 상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커서 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않는 경우에는 상기 하드웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 15】

제 14 항에 있어서, 상기 소프트웨어 블록의 출력 데이터는

상기 칩 설계 검증 프로그램으로부터 인가된 멀티 클럭에 응답하여 변화된

상기 소프트웨어 블록의 출력값과, 상기 소프트웨어 블록의 출력값이 변화될 때의 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값으로 구성되는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 16】

제 15 항에 있어서, 상기 유효한 소프트웨어 블록의 출력 데이터는

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 작은 경우에는 상기 소프트웨어 블록의 출력값이고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 커서 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 하드웨어 블록의 출력값이 변화하지 않는 경우에는 상기 소프트웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 17】

제 16 항에 있어서, 상기 소프트웨어측 동작 단계는

상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 상기 하드웨어의 출력 데이터를 수신하는 동작 진입 단계;

상기 칩 설계 검증 프로그램의 시스템 클럭 칩 설계 검증 프로그램의 시스템 클럭 값이 상기 인터페이스 수단의 시스템 클럭의 칩 설계 검증 프로그램의 시스템 클럭 값이 보다 작거나, 상기 칩 설계 검증 프로그램의 시스템 클럭 칩 설계 검증

프로그램의 시스템 클럭 값을 상기 수신한 인터페이스 수단의 시스템 클럭의 칩 설계 검증 프로그램의 시스템 클럭 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않으면, 상기 유효한 하드웨어 블록의 출력 데이터를 수신하였음을 확인하고 상기 소프트웨어 블록에 입력하는 입력 단계;

상기 칩 설계 검증 프로그램의 시스템 클럭 칩 설계 검증 프로그램의 시스템 클럭 값을 상기 수신한 상기 인터페이스 수단의 시스템 클럭의 칩 설계 검증 프로그램의 시스템 클럭 값과 동일하도록 증가시키기 이전에 상기 소프트웨어 블록의 출력값이 변화하면 상기 소프트웨어 블록의 출력 데이터를 상기 인터페이스 수단으로 전송하는 전송 단계; 및

상기 입력 단계 또는 상기 전송 단계가 완료되면, 상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 초기화하고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 모니터링하는 모니터링 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 18】

제 17 항에 있어서, 상기 입력 단계는

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같으면, 상기 수신한 하드웨어 블록의 출력값을 상기 유효한 하드웨어 블록의 출력 데이터로서 상기 소프트웨어 블록

에 입력하는 단계;

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 크면, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값보다 커질 때까지 상기 소프트웨어 블록의 출력값이 변화되지 않으면, 상기 수신한 하드웨어 블록의 출력값을 상기 유효한 하드웨어 블록의 출력 데이터로서 상기 소프트웨어 블록에 입력하는 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 19】

제 18 항에 있어서, 상기 입력 단계는

상기 칩 설계 검증 프로그램을 수행하여 상기 컴퓨터의 화면에 상기 유효한 하드웨어 블록의 출력 데이터를 디스플레이하는 단계를 더 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 20】

제 17 항에 있어서, 상기 전송 단계는

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 크면, 상기 칩 설계 검증 프로그램의 시스

템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값보다 작거나 같은 경우에 상기 소프트웨어 블록의 출력값이 변화되면, 상기 소프트웨어의 출력 데이터를 상기 인터페이스 수단에 전송하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 21】

제 20 항에 있어서, 상기 전송 단계는

상기 칩 설계 검증 프로그램을 수행하여 상기 컴퓨터의 화면에 상기 유효한 소프트웨어 블록의 출력 데이터를 디스플레이하는 단계를 더 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 22】

제 12 항에 있어서,

상기 소프트웨어 블록은 테스트 벤치를 구비하고, 상기 테스트 벤치는 상기 멀티 클럭 설정 값을 상기 칩 설계 검증 프로그램에 제공하고, 상기 테스트 벤치의 자체적인 멀티 클럭 대신에 상기 칩 설계 검증 프로그램의 멀티 클럭에 응답하여 동작하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 23】

제 16 항에 있어서, 상기 하드웨어측 동작 단계는

상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 발생되는 상기 소프트웨어의 출력 데이터를 수신하는 동작 진입 단계;

상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 보다 작거나, 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 인터페이스 수단의 출력값이 변화하지 않으면, 상기 유효한 소프트웨어 블록의 출력 데이터를 수신하였음을 확인하고 상기 하드웨어 블록에 입력하는 입력 단계;

상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시키기 이전에 상기 하드웨어 블록의 출력값이 변화하면 상기 하드웨어 블록의 출력 데이터를 상기 칩 설계 검증 프로그램으로 전송하는 전송 단계; 및

상기 입력 단계 또는 상기 전송 단계가 완료되면, 상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 초기화하고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 모니터링하는 모니터링 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 24】

제 23 항에 있어서, 상기 입력 단계는

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 작거나 같으면, 상기 수신한 소프트웨어 블록의 출력값을 상기 소프트웨어 블록의 유효 출력 데이터로서 상기 하드웨어 블록에 입력하는 단계;

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단이 시스템 클럭 카운트 값보다 크면, 상기 인터페이스 수단이 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 확인하는 단계;

상기 증가된 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커질 때까지 상기 하드웨어 블록의 출력값이 변화되지 않으면, 상기 수신한 소프트웨어 블록의 출력값을 상기 소프트웨어 블록의 유효 출력 데이터로서 상기 하드웨어 블록에 입력하는 입력 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 25】

제 24 항에 있어서, 상기 전송 단계는

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단이 시스템 클럭 카운트 값보다 크면, 상기 인터페이스 수단이 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같은 경우에 상기 하드웨어 블록의 출력값이 변화되면, 상기 하드웨어의 출력 데이터를 상기 인터페이스 수단에 전송하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 26】

적어도 하나 이상의 기능 블록을 구비하는 소프트웨어 블록 및 하드웨어 블록, 상기 소프트웨어 블록과 상기 하드웨어 블록의 동작을 검증하기 위한 칩 설계 검증 프로그램, 상기 칩 설계 검증 프로그램을 수행함에 의해 발생되는 상기 소프트웨어 블록의 입출력 데이터를 저장하는 저장부, 및 상기 하드웨어 블록과 상기 소프트웨어 블록간의 인터페이스를 수행하는 인터페이스 수단을 구비하는 칩 설계 검증 장치의 칩 설계 검증 방법에 있어서,

상기 칩 설계 검증 프로그램은 멀티 클럭 설정값을 획득하고, 상기 인터페이스 수단에 제공하고, 상기 멀티 클럭 설정값과 상기 칩 설계 검증 프로그램의 시스템 클럭에 따라 멀티 클럭을 발생하여 상기 소프트웨어 블록에 인가하고, 상기 인터페이스 수단은 상기 멀티 클럭 설정값과 상기 인터페이스 수단의 시스템 클럭에 따라 멀티 클럭을 발생하여 상기 하드웨어 블록에 인가하는 클럭 생성 단계;

상기 칩 설계 검증 프로그램의 멀티 클럭에 응답하여 동작하는 상기 소프트웨어 블록의 동작에 의해 발생된 출력데이터를 상기 인터페이스 수단으로

전송하고, 상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을

통해 수신된 상기 하드웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 유효한 상기 하드웨어 블록의 출력 데이터만을 상기 소프트웨어 블록에 인가하는 소프트웨어측 동작 단계; 및

상기 칩 설계 검증 프로그램의 멀티 클럭에 응답하여 동작하는 상기 하드웨어 블록의 동작에 의해 발생된 출력데이터를 상기 소프트웨어 블록으로 전송하고, 상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 수신되는 상기 소프트웨어 블록의 출력 데이터는 유효성 여부를 판단한 후, 유효한 상기 소프트웨어 블록의 출력 데이터만을 상기 하드웨어 블록에 인가하는 하드웨어측 동작 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 27】

제 26 항에 있어서, 상기 하드웨어 블록의 출력 데이터는

상기 인터페이스 수단으로부터 인가된 멀티 클럭에 응답하여 변화된 상기 하드웨어 블록의 출력값과, 상기 하드웨어 블록의 출력값이 변화될 때의 상기 인터페이스 수단의 시스템 클럭 카운트 값으로 구성되는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 28】

제 27 항에 있어서, 상기 유효한 하드웨어 블록의 출력 데이터는

상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작은 경우에는 상기 하드웨어 블록의 출력값이고,

상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커서 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않는 경우에는 상기 하드웨어 블록의 출력값인 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 29】

제 28 항에 있어서, 상기 소프트웨어 블록의 출력 데이터는

상기 칩 설계 검증 프로그램으로부터 인가된 멀티 클럭에 응답하여 변화된 상기 소프트웨어 블록의 출력값과, 상기 소프트웨어 블록의 출력값이 변화될 때의 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값으로 구성되는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 30】

제 29 항에 있어서, 상기 유효한 소프트웨어 블록의 출력 데이터는

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 작은 경우에는 상기 소프트웨어 블록의 출력값이고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단의 시스템 클럭 카운트 값보다 커서 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 하드웨어 블록의 출력값이 변화하지 않는 경우에는 상기 소프트웨어 블록

의 출력값인 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 31】

제 30 항에 있어서, 상기 소프트웨어측 동작 단계는

상기 칩 설계 검증 프로그램을 수행함에 의해서 상기 인터페이스 수단을 통해 상기 하드웨어의 출력 데이터를 수신하는 동작 진입 단계;

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 보다 작거나, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 소프트웨어 블록의 출력값이 변화하지 않으면, 상기 유효한 하드웨어 블록의 출력 데이터를 수신하였음을 확인하고 상기 소프트웨어 블록에 입력하는 입력 단계;

상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 상기 수신한 상기 인터페이스 수단의 시스템 클럭 카운트 값과 동일하도록 증가시키기 이전에 상기 소프트웨어 블록의 출력값이 변화하면 상기 소프트웨어 블록의 출력 데이터를 상기 인터페이스 수단으로 전송하는 전송 단계; 및

상기 입력 단계 또는 상기 전송 단계가 완료되면, 상기 증가된 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 초기화하고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 모니터링하는 모니터링 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방

법.

【청구항 32】

제 31 항에 있어서, 상기 입력 단계는

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같으면, 상기 수신한 하드웨어 블록의 출력값을 상기 유효한 하드웨어 블록의 출력 데이터로서 상기 소프트웨어 블록에 입력하는 단계;

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 크면, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값보다 커질때까지 상기 소프트웨어 블록의 출력값이 변화되지 않으면, 상기 수신한 하드웨어 블록의 출력값을 상기 유효한 하드웨어 블록의 출력 데이터로서 상기 소프트웨어 블록에 입력하는 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 33】

제 32 항에 있어서, 상기 입력 단계는

상기 칩 설계 검증 프로그램을 수행하여 상기 컴퓨터의 화면에 상기 하드웨

어 블록의 출력 데이터를 디스플레이하는 단계를 더 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 34】

제 31 항에 있어서, 상기 전송 단계는

상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값이 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 크면, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 수신한 인터페이스 수단의 시스템 클럭 카운트 값보다 작거나 같은 경우에 상기 소프트웨어 블록의 출력값이 변화되면, 상기 소프트웨어의 출력 데이터를 상기 인터페이스 수단에 전송하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 35】

제 34 항에 있어서, 상기 전송 단계는

상기 칩 설계 검증 프로그램을 수행하여 상기 컴퓨터의 화면에 상기 유효한 소프트웨어 블록의 출력 데이터를 디스플레이하는 단계를 더 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 36】

제 26 항에 있어서,

상기 소프트웨어 블록은 테스트 벤치를 구비하고, 상기 테스트 벤치는 상기 멀티 클럭 설정 값을 상기 칩 설계 검증 프로그램에 제공하고, 상기 테스트 벤치의 자체적인 멀티 클럭 대신에 상기 칩 설계 검증 프로그램의 멀티 클럭에 응답하여 동작하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 37】

제 30 항에 있어서, 상기 하드웨어측 동작 단계는

상기 인터페이스 수단에서 상기 칩 설계 검증 프로그램을 수행함에 의해서 발생되는 상기 소프트웨어의 출력 데이터를 수신하는 동작 진입 단계;

상기 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 보다 작거나, 상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시켜도 상기 인터페이스 수단의 출력값이 변화하지 않으면, 상기 유효한 소프트웨어 블록의 출력 데이터를 수신하였음을 확인하고 상기 하드웨어 블록에 입력하는 입력 단계;

상기 인터페이스 수단의 시스템 클럭 카운트 값을 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값과 동일하도록 증가시키기 이전에 상기 하드웨어 블록의 출력값이 변화하면 상기 하드웨어 블록의 출력 데이터를 상기 칩 설계 검증 프로그램으로 전송하는 전송 단계; 및

상기 입력 단계 또는 상기 전송 단계가 완료되면, 상기 증가된 칩 설계 검증

프로그램의 시스템 클럭 카운트 값을 초기화하고, 상기 칩 설계 검증 프로그램의 시스템 클럭 카운트 값을 증가시키면서 상기 소프트웨어 블록의 출력값이 변화되는지를 모니터링하는 모니터링 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 38】

제 37 항에 있어서, 상기 입력 단계는

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단이 시스템 클럭 카운트 값보다 작거나 같으면, 상기 수신한 소프트웨어 블록의 출력값을 상기 유효한 소프트웨어 블록의 출력 데이터로서 상기 하드웨어 블록에 입력하는 단계;

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단이 시스템 클럭 카운트 값보다 크면, 상기 인터페이스 수단이 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

상기 증가된 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 커질 때까지 상기 하드웨어 블록의 출력값이 변화되지 않으면, 상기 수신한 소프트웨어 블록의 출력값을 상기 유효한 소프트웨어 블록의 출력 데이터로서 상기 하드웨어 블록에 입력하는 단계를 구비하는 것을 특징으로 하는 칩 설계 검증 방법.

【청구항 39】

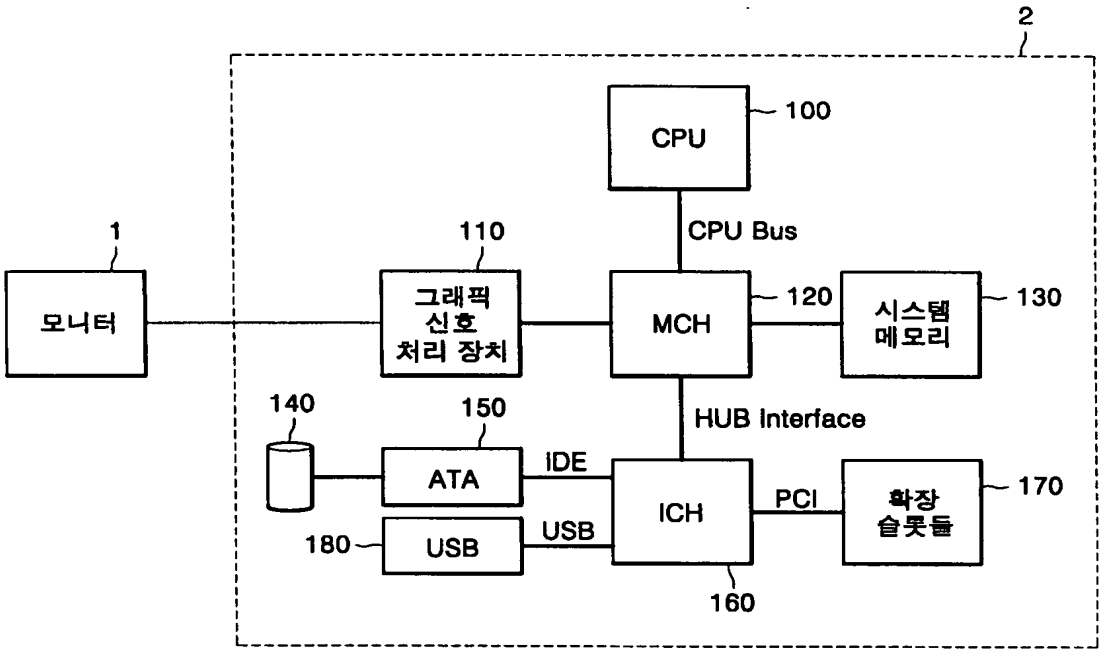
제 37 항에 있어서, 상기 전송 단계는

상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값이 상기 인터페이스 수단이 시스템 클럭 카운트 값보다 크면, 상기 인터페이스 수단이 시스템 클럭 카운트 값을 증가시키면서 상기 하드웨어 블록의 출력값이 변화되는지를 확인하는 단계; 및

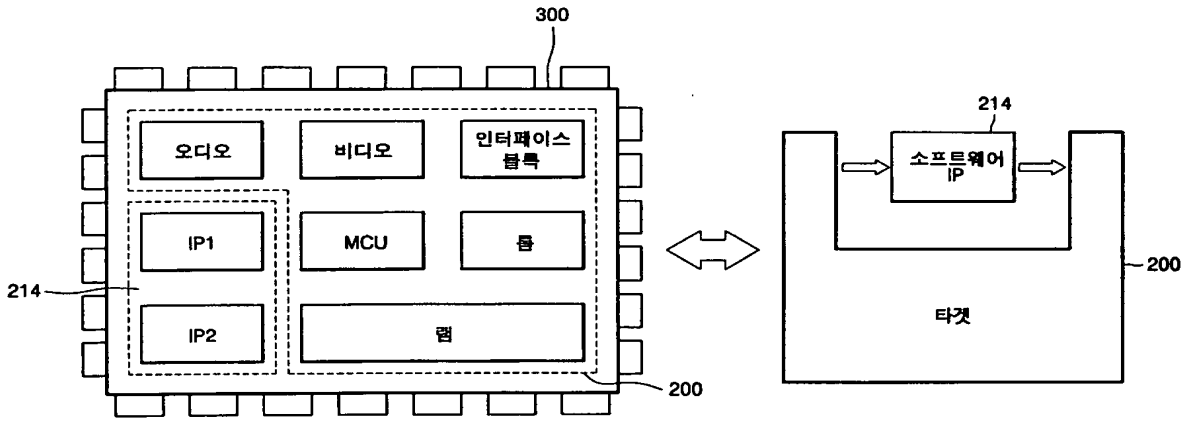
상기 증가된 인터페이스 수단의 시스템 클럭 카운트 값이 상기 수신한 칩 설계 검증 프로그램의 시스템 클럭 카운트 값보다 작거나 같은 경우에 상기 하드웨어 블록의 출력값이 변화되면, 상기 하드웨어의 출력 데이터를 상기 인터페이스 수단에 전송하는 것을 특징으로 하는 칩 설계 검증 방법.

【도면】

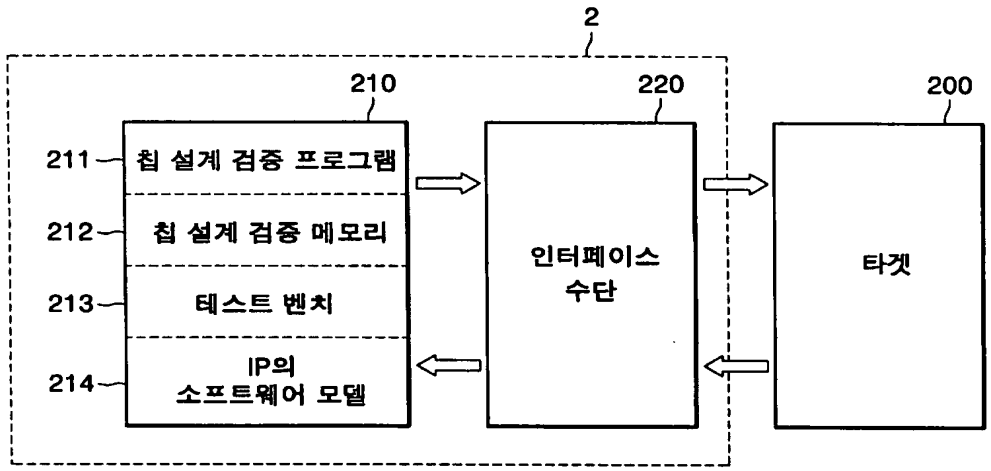
【도 1】


2
CPU
100
CPU Bus
1
110
모니터
그래픽
신호
처리 장치
MCH
120
시스템
메모리
130
140
150
HUB Interface
ATA
IDE
ICH
PCI
확장
슬롯들
170
180
USB
USB
160

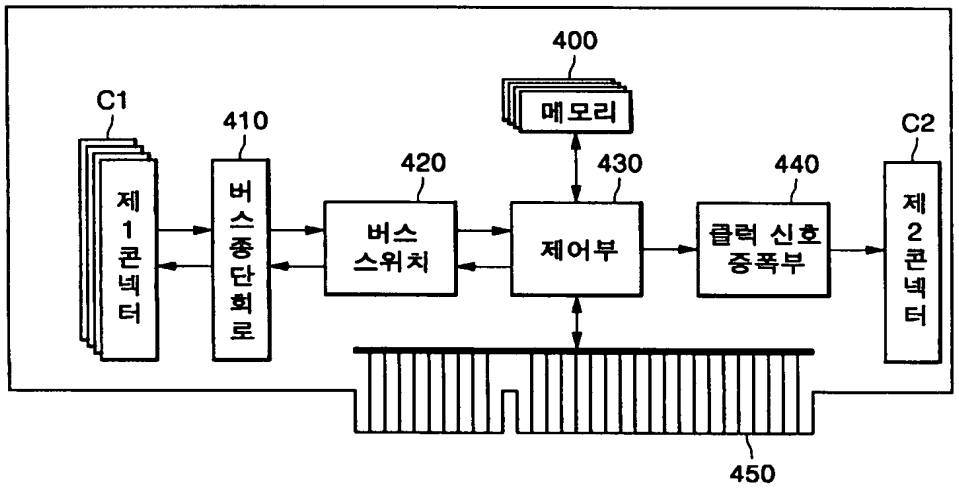
【도 2】


300
오디오
비디오
인터페이스
블록
IP1
MCU
롬
214
IP2
램
200
214
소프트웨어
IP
200
타겟

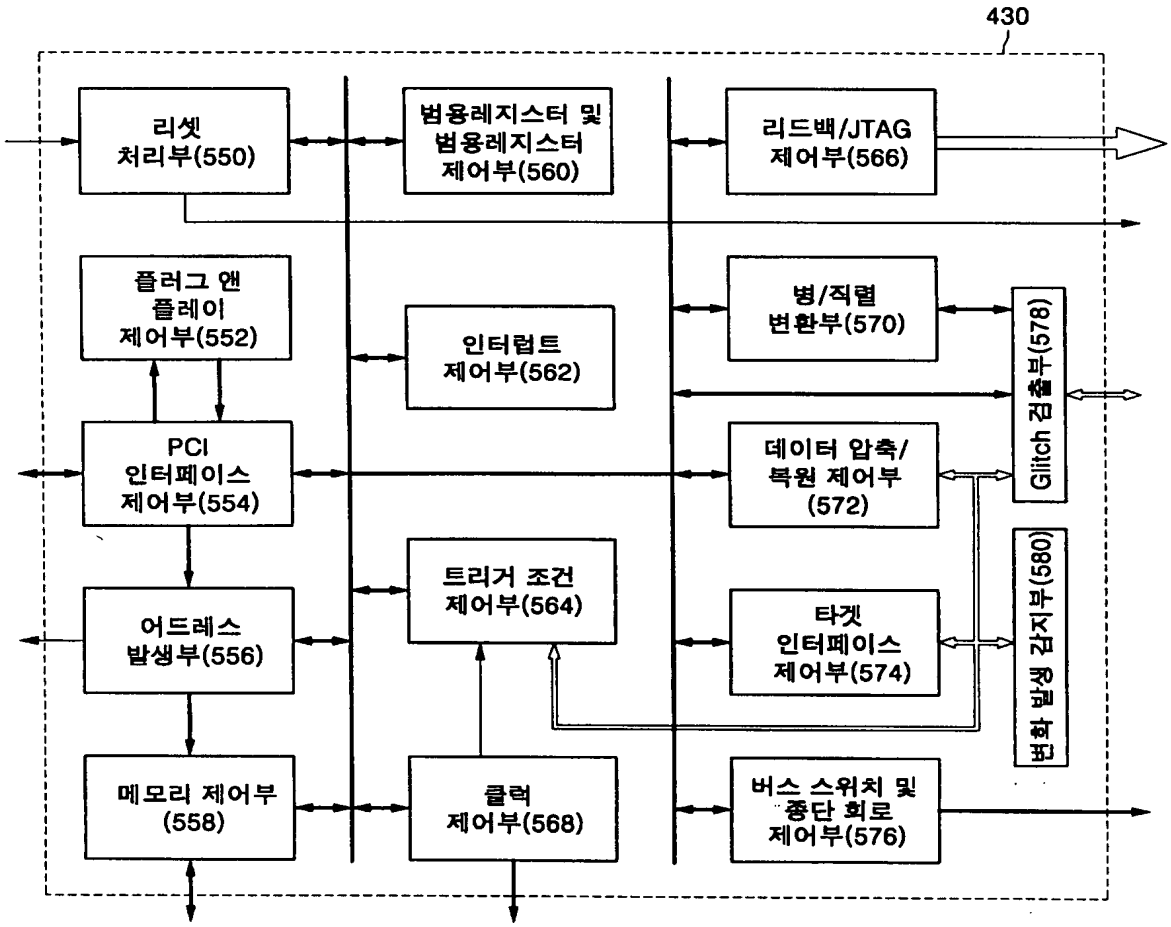
【도 3】


2
210
220
200
211
칩 설계 검증 프로그램
212
칩 설계 검증 메모리
213
테스트 벤치
214
IP의
소프트웨어 모델
인터페이스
수단
타겟

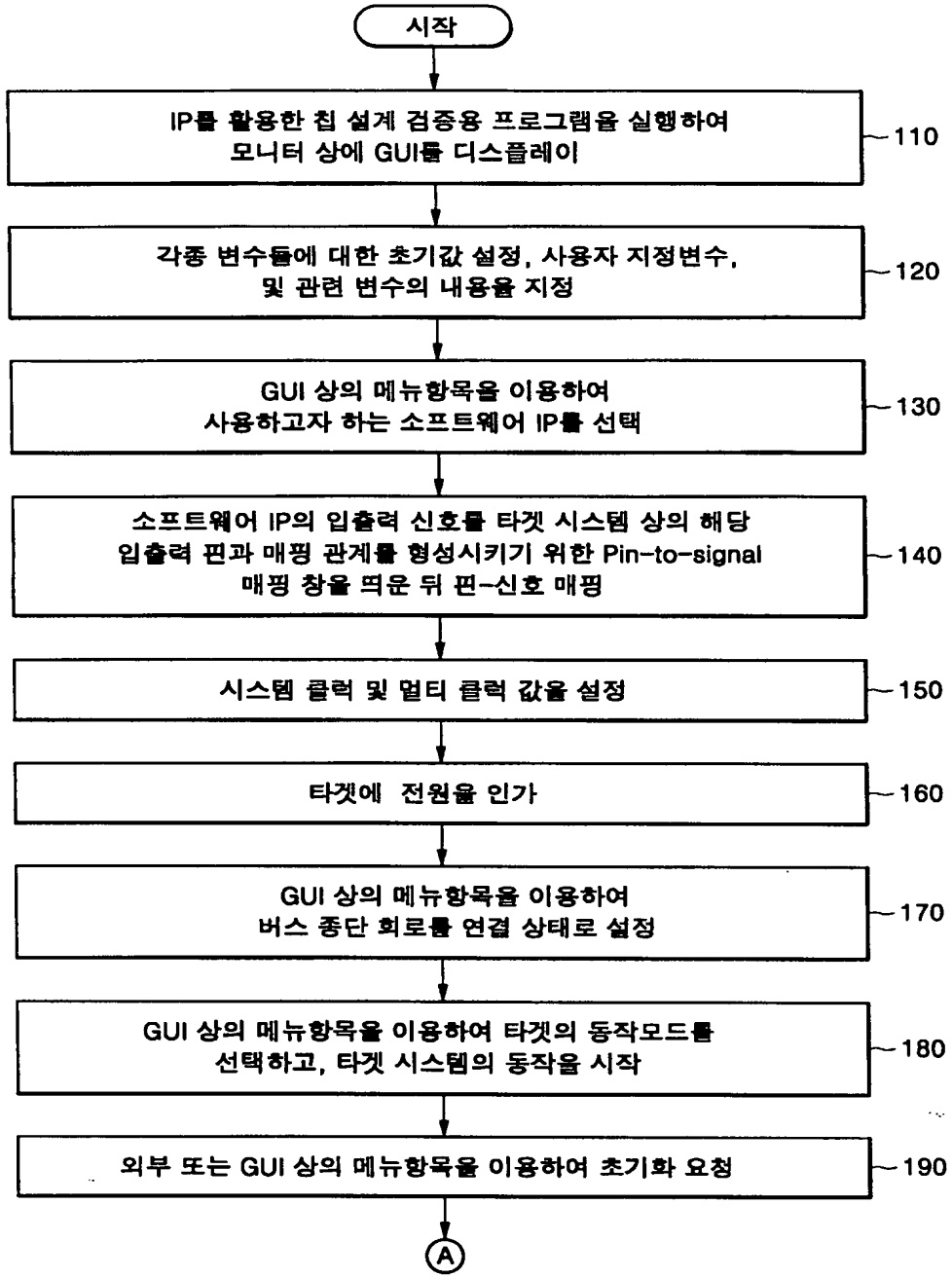
【도 4】


400
메모리
C1
410
420
430
440
C2
제
1
콘
넥
터
버
스
종
단
회
로
버스
스위치
제어부
클럭 신호
증폭부
제
2
콘
넥
터
450

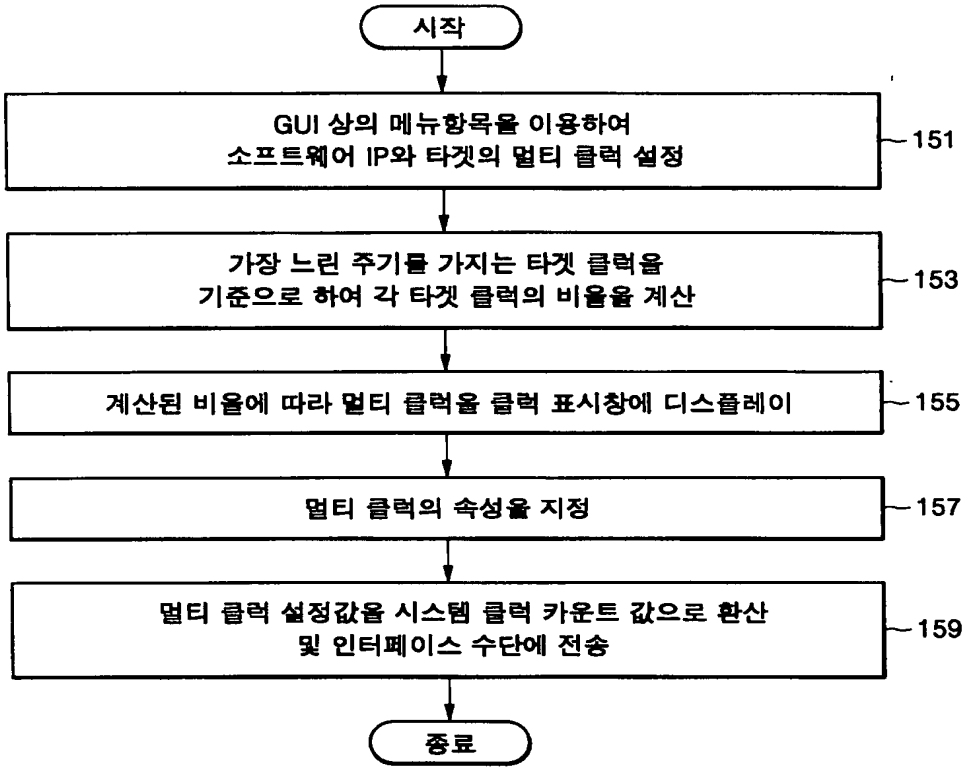
【도 5】


430
리셋 처리부(550)
범용레지스터 및 범용레지스터 제어부(560)
리드백/JTAG 제어부(566)
플러그 앤 플레이 제어부(552)
인터럽트 제어부(562)
병/직렬 변환부(570)
Glitch 검출부(578)
PCI 인터페이스 제어부(554)
데이터 압축/복원 제어부(572)
트리거 조건 제어부(564)
변화 발생 감지부(580)
어드레스 발생부(556)
타겟 인터페이스 제어부(574)
메모리 제어부(558)
클럭 제어부(568)
버스 스위치 및 종단 회로 제어부(576)

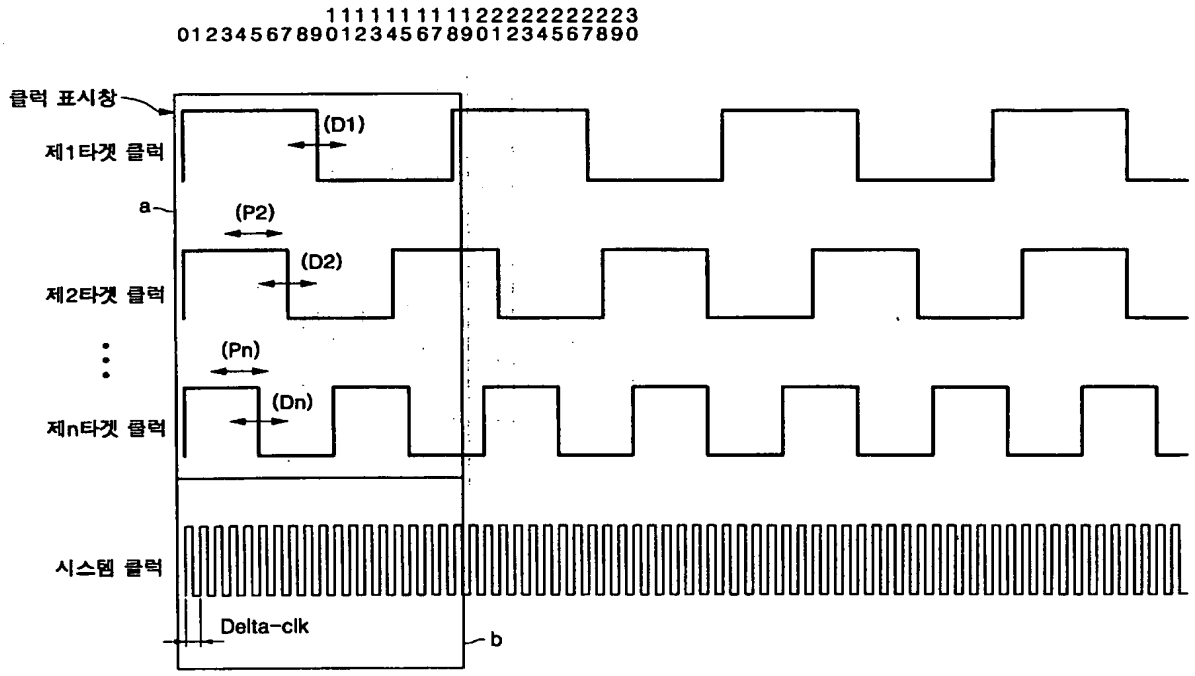
【도 6】


시작
IP를 활용한 칩 설계 검증용 프로그램을 실행하여 모니터 상에 GUI를 디스플레이
110
각종 변수들에 대한 초기값 설정, 사용자 지정변수, 및 관련 변수의 내용을 지정
120
GUI 상의 메뉴항목을 이용하여 사용하고자 하는 소프트웨어 IP를 선택
130
소프트웨어 IP의 입출력 신호를 타겟 시스템 상의 해당 입출력 핀과 매핑 관계를 형성시키기 위한 Pin-to-signal 매핑 창을 띄운 뒤 핀-신호 매핑
140
시스템 클럭 및 멀티 클럭 값을 설정
150
타겟에 전원을 인가
160
GUI 상의 메뉴항목을 이용하여 버스 종단 회로를 연결 상태로 설정
170
GUI 상의 메뉴항목을 이용하여 타겟의 동작모드를 선택하고, 타겟 시스템의 동작을 시작
180
외부 또는 GUI 상의 메뉴항목을 이용하여 초기화 요청
190
A

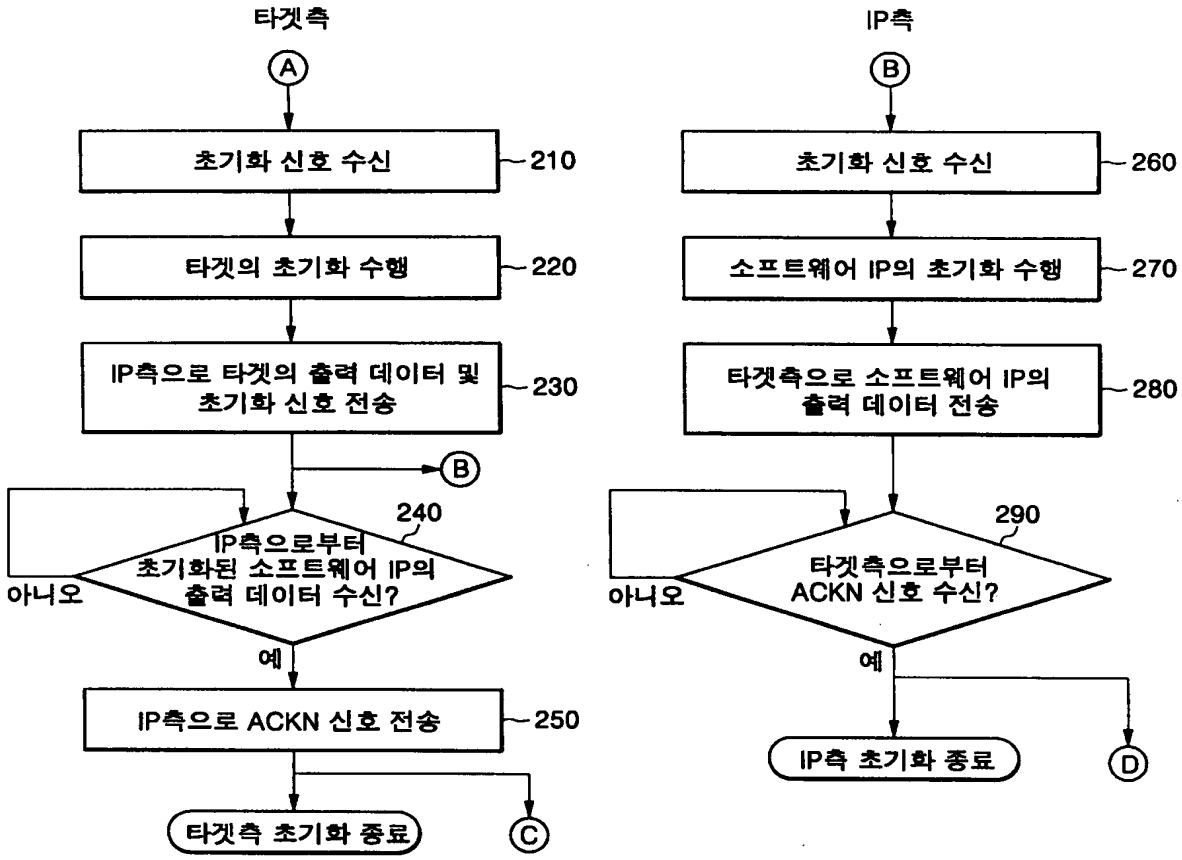
【도 7a】


시작
GUI 상의 메뉴항목을 이용하여
소프트웨어 IP와 타겟의 멀티 클럭 설정
151
가장 느린 주기를 가지는 타겟 클럭을
기준으로 하여 각 타겟 클럭의 비율을 계산
153
계산된 비율에 따라 멀티 클럭을 클럭 표시창에 디스플레이
155
멀티 클럭의 속성을 지정
157
멀티 클럭 설정값을 시스템 클럭 카운트 값으로 환산
및 인터페이스 수단에 전송
159
종료

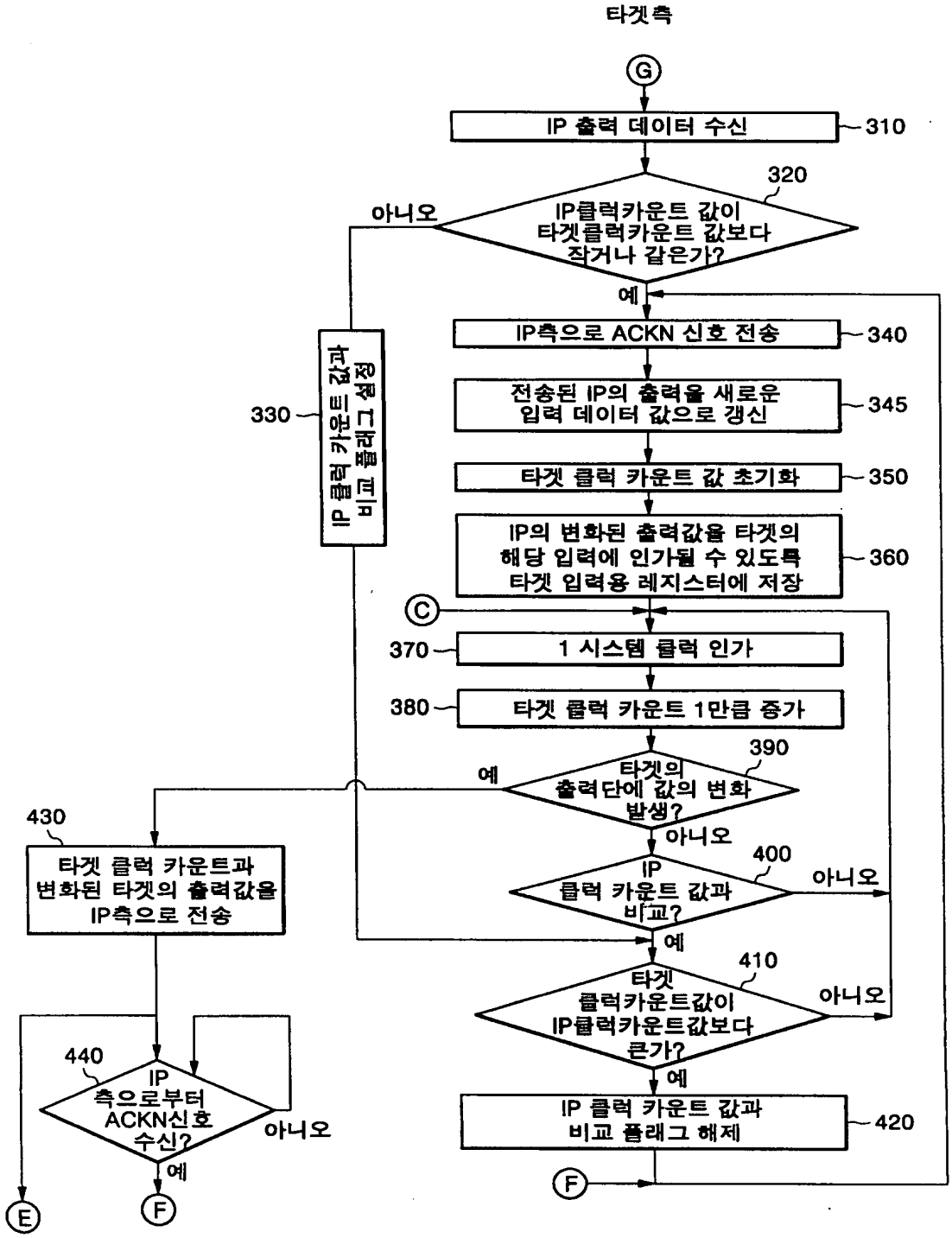
【도 7b】


1111111111222222222223
01234567890123456789012345678 90
클럭 표시창
제1타겟 클럭
(D1)
a
(P2)
제2타겟 클럭
(D2)
(Pn)
제n타겟 클럭
(Dn)
시스템 클럭
Delta-clk
b

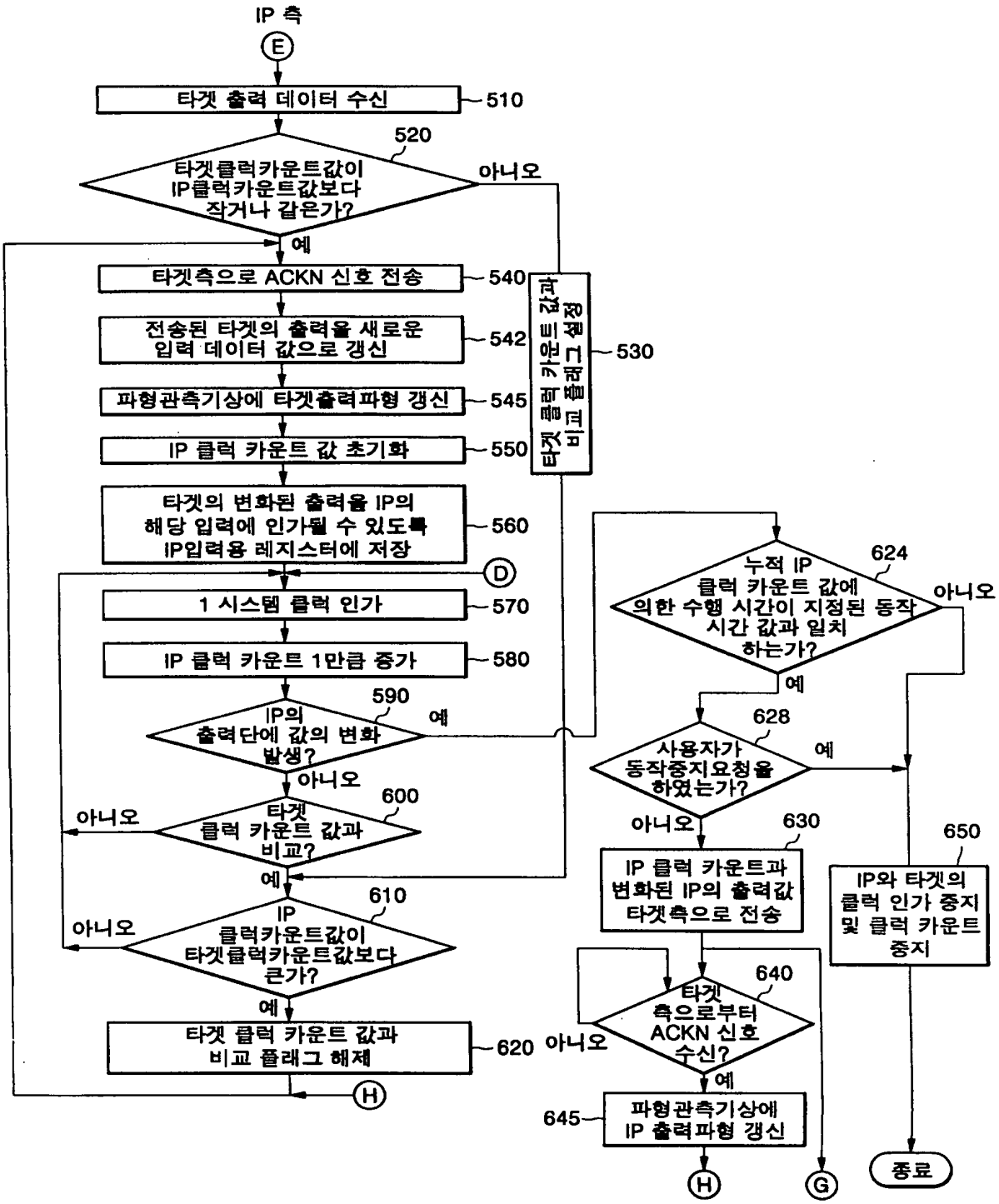
【도 8】


타겟측
A
초기화 신호 수신
210
타겟의 초기화 수행
220
IP측으로 타겟의 출력 데이터 및 초기화 신호 전송
230
B
240
IP측으로부터 초기화된 소프트웨어 IP의 출력 데이터 수신?
아니오
예
IP측으로 ACKN 신호 전송
250
타겟측 초기화 종료
C
IP측
B
초기화 신호 수신
260
소프트웨어 IP의 초기화 수행
270
타겟측으로 소프트웨어 IP의 출력 데이터 전송
280
290
타겟측으로부터 ACKN 신호 수신?
아니오
예
IP측 초기화 종료
D

【도 9a】


타겟측
G
IP 출력 데이터 수신
310
320
IP클럭카운트 값이 타겟클럭카운트 값보다 작거나 같은가?
아니오
예
IP측으로 ACKN 신호 전송
340
전송된 IP의 출력을 새로운 입력 데이터 값으로 갱신
345
타겟 클럭 카운트 값 초기화
350
IP의 변화된 출력값을 타겟의 해당 입력에 인가될 수 있도록 타겟 입력용 레지스터에 저장
360
330
IP 클럭 카운트 값과 비교 플래그 설정
C
370
1 시스템 클럭 인가
380
타겟 클럭 카운트 1만큼 증가
390
타겟의 출력단에 값의 변화 발생?
예
아니오
400
IP 클럭 카운트 값과 비교?
아니오
예
410
타겟 클럭카운트값이 IP클럭카운트값보다 큰가?
아니오
예
IP 클럭 카운트 값과 비교 플래그 해제
420
F
430
타겟 클럭 카운트과 변화된 타겟의 출력값을 IP측으로 전송
440
IP 측으로부터 ACKN신호 수신?
아니오
예
E
F

【도 9b】


IP 측
E
타겟 출력 데이터 수신
510
520
타겟클럭카운트값이 IP클럭카운트값보다 작거나 같은가?
아니오
예
타겟측으로 ACKN 신호 전송
540
전송된 타겟의 출력을 새로운 입력 데이터 값으로 갱신
542
파형관측기상에 타겟출력파형 갱신
545
IP 클럭 카운트 값 초기화
550
타겟의 변화된 출력을 IP의 해당 입력에 인가될 수 있도록 IP입력용 레지스터에 저장
560
타겟 클럭 카운트 값과 비교 플래그 설정
530
D
1 시스템 클럭 인가
570
IP 클럭 카운트 1만큼 증가
580
590
IP의 출력단에 값의 변화 발생?
예
아니오
600
타겟 클럭 카운트 값과 비교?
아니오
예
610
IP 클럭카운트값이 타겟클럭카운트값보다 큰가?
아니오
예
타겟 클럭 카운트 값과 비교 플래그 해제
620
H
624
누적 IP 클럭 카운트 값에 의한 수행 시간이 지정된 동작 시간 값과 일치 하는가?
아니오
예
628
사용자가 동작중지요청을 하였는가?
예
아니오
630
IP 클럭 카운트과 변화된 IP의 출력값 타겟측으로 전송
650
IP와 타겟의 클럭 인가 중지 및 클럭 카운트 중지
640
타겟 측으로부터 ACKN 신호 수신?
아니오
예
645
파형관측기상에 IP 출력파형 갱신
H
G
종료